OpenH323 Gatekeeper - El GNU Gatekeeper

Este manual es mantenido y actualizado por: *Jan Willamowius <jan@willamowius.de>*

Este es el manual de usuario de cómo compilar, instalar, configurar y monitorear el OpenH323 Gatekeeper - El GNU Gatekeeper.

Índice

1. Introducción **3**

1.1. Acerca de . . . 3
1.2. Copyright . . . 4
1.3. Nombre . . . 4
1.4. Características . . . 4
1.5. Download . . . 7
1.6. Listas de correo . . . 7
1.7. Contribuidores . . . 7

2. Compilacion e Instalacion **8**

2.1. Compilando el Gatekeeper . . . 8
2.2. Instalacion del Gatekeeper . . . 9
2.3. Binarios pre-compilados . . . 9

3. Empezando(Tutorial) **9**

3.1. Un simple primer experimento . . . 9
3.2. Usando la interface de Status para monitorizar el gatekeeper . . . 10
3.3. Arrancando el gatekeeper en modo enrutado. . . . 10
3.4. Una centralita (PBX) virtual: desconectando llamadas . . . 11
3.5. Enrutando llamadas sobre un gateway para alcanzar usuarios externos . . . 11
3.6. Reescribiendo numero E.164 . . . 12

4. Configuración Básica del Gatekeeper **13**

4.1. Opciones de la Línea de Comandos . . . 13
4.1.1. Lo Básico . . . 13
4.1.2. Modo del Gatekeeper . . . 14
4.1.3. Información de Depuración . . . 14
4.2. Archivo de Configuración . . . 14
4.2.1. Sección [Gatekeeper::Main] . . . 15
4.2.2. Sección [GkStatus::Auth] . . . 19

4.2.3. Sección [LogFile] . . . 19
4.2.4. Sección [RoutedMode] . . . 20
4.2.5. Sección [Proxy] . . . 23

5. Configuración de Enrutado **25**
5.1. Sección [Políticas de enrutado] . . . 25
5.2. Sección [RasSrv::RewriteE164] . . . 26
5.3. Sección [RasSrv::GWRewriteE164] . . . 26
5.4. Sección [Endpoint::RewriteE164] . . . 27

6. Configuración RAS **28**
6.1. Sección [RasSrv::GWPrefixes] . . . 28
6.2. Sección [RasSrv::PermanentEndpoints] . . . 28
6.3. Sección [RasSrv::RRQFeatures] . . . 29
6.4. Sección [RasSrv::ARQFeatures] . . . 29
6.5. Section [NATedEndpoints] . . . 30

7. Configuración de la Autenticación **30**
7.1. Sección [Gatekeeper::Auth] . . . 30
7.2. Sección [SimplePasswordAuth] . . . 32
7.3. Sección [SQLPasswordAuth] . . . 32
7.4. Sección [RasSrv::RRQAuth] . . . 33
7.5. Sección [SQLAliasAuth] . . . 34
7.6. Sección [PrefixAuth] . . . 35
7.7. Sección [RadAuth] . . . 36
7.8. Section [RadAliasAuth] . . . 38

8. Configuración de Accounting **41**
8.1. Sección [Gatekeeper::Acct] . . . 41
8.2. Sección [FileAcct] . . . 44
8.3. Sección [RadAcct] . . . 46
8.4. Sección [SQLAcct] . . . 47

9. Configuración de Vecinos **49**
9.1. Sección [RasSrv::Neighbors] . . . 49
9.2. Sección [RasSrv::LRQFeatures] . . . 51
9.2.1. Sección [Neighbor::...] . . . 52

10.Configuración Por-Terminal **53**

10.1. Sección [EP::...] 53

11.Configuración Avanzada **54**

11.1. Sección [CallTable] 54

11.2. Sección [Endpoint] 54

11.3. Sección [CTI::Agents] 56

11.4. Sección [SQLConfig] 57

12.Monitoreando el Gatekeeper **60**

12.1. Puerto de Estado 60

12.1.1. Areas de aplicación 61

12.1.2. Ejemplos: 61

12.1.3. GUI para el gatekeeper 62

12.2. Comandos (Referencia) 63

12.3. Mensajes (Referencia) 68

1. Introducción

1.1. Acerca de

OpenH323 Gatekeeper - El GNU Gatekeeper <http://www.gnugk.org/> es un proyecto de codigo abierto que implementa un gatekeeper H.323. Un gatekeeper ofrece de servicios de control de llamada a los clientes H.323. Es una parte integral de las instalaciones telefónicas en internet más utiles que están basadas en el estándar H.323.

De acuerdo con la Recomendacion H.323, un gatekeeper deberá ofrecer los siguientes servicios:

- Traduccion de direcciones
- Control de admisiones
- Control del ancho de banda
- Gestion de zona
- Señalizacion del control de llamada
- Autorizacion de llamada
- Gestion del ancho de banda
- Gestion de llamada

El GNU Gatekeeper implementa la mayor parte de estas funciones basandose en la pila de protocolos *OpenH323* <http://www.openh323.org/>

La recomendacion H.323 es un estandar internacional publicado por el *ITU* <http://www.itu.int/>. Es un estandar de comunicaciones para audio, video y datos a traves de Internet. Consultar tambien "Paul Jones"*primer on H.323* <http://www.packetizer.com/iptel/h323/>.

Para una descripcion detallada de lo que hace gatekeeper consulte *here* <http://www.webproforum.com/h323/topic06.html>.

1.2. Copyright

Esta publicado bajo la licencia *GNU General Public License* (GNU GPL). Ademas, se garantiza explicitamente el derecho a enlazar este codigo a las librerias OpenH323 y OpenSSL.

Generalmente hablando, la GNU GPL le permite copiar, distribuir, revender o modificar este software, pero requiere que los trabajos derivados sean publicados bajo la GNU GPL. Esto significa que debe publicar todo el codigo fuente de todas las extensiones al gatekeeper y de todos los programas que donde usted incluya el gatekeeper. Consulte el fichero COPYING para mas detalles.

Si no es esto lo que usted quiere, debe crear una interface con el gatekeeper a traves del puerto de estado y comunicarse con éste via TCP. De esta forma usted solo tiene que integrar la funcionalidad basica dentro del gatekeeper (y facilitar el codigo fuente de esta) y puede mantener otras partes de su aplicacion bajo otra licencia.

1.3. Nombre

El nombre formal de este proyecto es *OpenH323 Gatekeeper - El GNU Gatekeeper*, abreviado en *GnuGk*. Por favor no lo confunda con otros proyectos de gatekeeper.

Hay muchos proyectos de gatekeepers open-source basados en la pila del protocol OpenH323.

- *OpenGatekeeper* <http://opengatekeeper.sourceforge.net/> - por *Egoboo* <http://www.egoboo.com/>

  Un completo gatekeeper disponible gratuitamente bajo MPL.Este proyecto ha estado inactivo por un tiempo. Existe un proxy H.323 basado en OpenGatekeeper, consulte *OpenH323Proxy* <http://openh323proxy.sourceforge.net/>.

- *OpenGK* <http://www.openh323.org/code.html> - por *Equivalence* <http://www.equival.com.au/>

  Solo en fase inicial.

- OpenH323 Gatekeeper - este mismo.

El tener diferentes gatekeepers con nombres muy similares es confuso para la mayoria de los usuarios. Como nuestro .ºpenH323 Gatekeeper"fue el primero en aparecer, no es nuestra culpa que otros hayan elegido nombres similares. Pero para que la distincion sea un poco mas clara sin confundir a la gente todavia mas hemos decidido darle un subtitulo al proyecto .ºpenH323 Gatekeeper - El GNU Gatekeeperz empezar a usar `gnugk` como nombre para los ejecutables.

1.4. Características

La version 2.2.2 contiene las siguientes novedades y correcciones:

- Se han quitado los modulos obsoletos de autentificacion MySQLAliasAuth y MySQLPasswordAuth.
- Los modulos de SQL aceptan solo un database host en el parametro `Host`.

La version 2.2.1 cotiene las siguientes novedades y correcciones:

- Mejorados los prefijos de sustitucion para las politicas de enrutamiento. Un punto (.) sustituye cualquier digito.
- Mejorados los prefijos se sustitucion para vecinos. Un punto (.) sustituye cualquier digito, ! al inicio desactiva el prefijo.
- Un error de configuracion durante la recarga de la configuracion hace que el gatekeeper falle.
- Seleccion de puertos mas fiable para rangos de puertos del Q.931, H.245, T.120 y RTP (anteriormente, una recarga de la configuracion podia causar muchas llamanadas fallidas a causa de la imposibilidad de asignarles un nuevo socket).
- Ahora se deja al SO escojer el numero de puerto por defecto para RTPPortRange.
- Las reglas de reescritura son mas flexibles (tanto globales como para-gateway) con nuevos metacaracteres ('.' y '%').
- Mejorados los prefijos de sustitucion para gateways. Un punto (.) sustituye cualquier digito, un ! al principio desactiva el prefijo.
- Si las opciones correspondientes `Screen...` estan activadas se inserta el Calling-Party-Number-IE/Display-IE perdido.
- El gatekeeper se cerrara si hay errores en la configuracion de los modulos de SQL auth/acct.
- El tipo de numero Called-Station-Id puede ser seleccionado entre el original (el numero marcado) y el que ha sido reescrito. Nueva opcion `UseDialedNumber` a configurar para los modulos 7.7 (RadAuth)/7.8 (RadAliasAuth)/8.3 (RadAcct), nueva variable %{Dialed-Number} para los modulos 8.4 (SQLAcct) y 8.2 (FileAcct).
- Se posibilita la personalizacion del formato de fecha. Nuevas variables `TimestampFormat` a configurar para las secciones principal, 8.4 ([SqlAcct]), 8.3 ([RadAcct]), 8.2 ([FileAcct]) y 11.1 ([CallTable]).
- Los modulos RadAuth/RadAliasAuth pueden añadir/eliminar alias de endpoint durante el registro endpoint (usando h323-ivr-in=terminal-alias: Cisco AV-Pair).
- Nueva opcion `TcpKeepAlive` para solventar el problema con los errores de red y llamadas colgando en la tabla de llamada. Consulte docs/keepalive.txt para mas detalles.
- Nuevo comando para mostrar el estado del puerto `RouteToGateway`.

La version 2.2.0 cotiene las siguientes novedades y correcciones:

- Nueva opcion de configuracion `RoundRobinGateways`.
- Limites de capacidad de llamada y prioridad de routing para gateways. Nuevas secciones de configuracion `EP::` para parametros de configuracion por-endpoint (consulte 10 (configuracion Por-Endpoint)).
- El tratamiento de proxy RTP se ha movido a hilos separados para proxy RTP, asi el procesamiento de los mensajes de señalizacion no bloquea los paquetes RTP. Nueva opcion de configuracion `RtpHandlerNumber`.
- La opcion REUSE_ADDRESS ha sido activada para sockets de escucha en el modo non-LARGE_FDSET para solucionar un fallo que desabilitaba al gatekeeper para poder abrirp purestos de escucha despues de reiniciar.

- Habilidad para establecer el destino de la llamada en los modulos de autentificacion. Enrutamiento de llamada basado en RADIUS.
- Soporte para importar la tabla de tarifas de SqlBill desde una hoja de calculo OpenOffice.org.
- Solucionada la gestion del campo sourceInfo LRQ -ahora contiene un identificador H.323 del gatekeeper. Los datos no estandar y campos gatekeeperIdentifier son establecidos solo cuando la otra parte es definida como GnuGk.
- Habilidad para establecer secretos compartidos para cada servidor radius de forma separada.
- Nueva, mucho mas rapida, implementacion del cliente para Radius.
- Si la reescritura de Called-Party-Number-IE ocurrio demasiado tarde, los modulos auth/acct reciben el numero original en vez del reescrito.
- Arreglado el proxying de paquetes RTP, asi los sockets RTP no estan cerrados en errores temporales (como el socket remoto no esta listo todavia). Este bug affecta especialmente al NAT transversal y de situacion, cuando se envia audio demasiado pronto, cuando el camido del proxy inverso no ha sido todavia establecido.
- Arreglada la manipulacion de RRJ desde un GnuGk alternativo.
- Nuevo modulo de gestion de cuentas SQL directo (8.4 ([SQLAcct])).
- Arreglada la manipulacion de multiples mensajes de respuesta (RIP/LCF/LRJ) desde vecinos.
- Soporte para CallCreditServiceControl en mensajes RCF y ACF, que permite notificar el limite de duracion de la llamada y el balance de la cuenta de usuario a los endpoints. Actualmente los modules RadAuth y RadAliasAuth soportan esta caracteristica.
- Rotacion del fichero log, nueva seccion de configuracion `LogFile`, nuevos comandos de estado de la interface `setlog` y `rotatelog`.
- No se incluye un token de acceso inválido (con identificadores de objeto nulos) en los mensajes LCF para prevenir problemas de interoperabilidad.
- Mejor manejo de llamadas multiples sobre un unico canal de señalizacion mediante el establecimiento de los campos multipleCalls y maintainConnection H.225.0 a FALSE en todos los mensajes que pasan a traves del gatekeeper.
- Mejor manipulacion de User-Name, Calling-Station-Id y Called-Station-Id.
- La bandera IncludeEndpointIP para RadAuth, RadAliasAuth y RadAcct queda obsoleta, estos modulos enviaran siempre Framed-IP-Address.
- Nueva SetupUnreg bandera Gatekeeper::Auth para fijar que la autentificacion en Q.931 Setup solo este disponible para endpoints no registrados.
- Nuevo parametro h323-ivr-out=h323-call-id para RADIUS que contiene un identificador de llamada H.323.
- El sistema de facturacion SQL de la seccion de contribuciones puede autentificar ahora usuario con soloamente su IP (ignorando el nombre de usuario) y tiene un nuevo y mas flexible motor de tarifas/valoracion.
- RadAliasAuth ahora puede autentificar mensajes de Setup sin que tenga que estar el campo sourceAddress presente (usando en tal caso usara Calling-Party-Number).

- Mejor manipulacion de la señal para prevenir cierres accidentales del gatekeeper (debido a SIGPIPE, por ejemplo).
- La rotacion CDR por numero de lineas fuciona correctamente.

1.5. Download

En *la pagina de descargas* <http://www.gnugk.org/h323download.html> estan disponibles la ultima version estable y una version de desarrollo.

La ultima version del codigo fuente esta en el CVS en *Sourceforge* <http://sourceforge.net/cvs/?group_id=4797> (*Web-GUI* <http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/openh323gk/>). Recuerde - Esta version puede contener muchos fallos.

Tambien puede descargar algunos ejecutables desde *la pagina de descargas* <http://www.gnugk.org/h323download.html>. Solo se ponen a disposicion los ejecutables de algunas versiones.

1.6. Listas de correo

Existen dos listas de correo para el proyecto, una para los desarrolladores y otra para los usuarios.

Las preguntas de los usuarios deben de ser enviadas a *users mailing list* <mailto:Openh323gk-users@sourceforge.net>. Puede encontrar un archivo de la lista de correo pulse *aqui* <http://sourceforge.net/mailarchive/forum.php?forum_id=8549>. Para apuntarse a la lista de correo, pulse *aqui* <http://lists.sourceforge.net/lists/listinfo/openh323gk-users>.

Para notificar problemas o enviar bugs/parches, envie los mails a la *lista para desarrolladores* <mailto:Openh323gk-developer@sourceforge.net>. El archivo de la lista esta *aqui* <http://sourceforge.net/mailarchive/forum.php?forum_id=3079>. Porfavor envie las preguntas de usuario a la lista para usuarios y deje esta lista solo para desarrolladores! Si desea contribuir al proyecto, porfavor *unase a la lista* <http://lists.sourceforge.net/lists/listinfo/openh323gk-developer>.

Nota: Porfavor no envie preguntas a los mails privados de los desarrolladores. Normalmente estamos muy ocupados. **no** nos gustaria ser su asesor privado, a no ser que desee pagarnos. Envie sus problemas a la lista de correo pública apropiada para que todo el mundo pueda ayudarle.

Porfavor, no envie tampoco los problemas especificos de GnuGk a la lista de OpenH323, o vice versa. Son proyectos diferentes, aunque muy relacionados.

Antes de enviar un email, asegurese de que ha leido los documentos relaciondos detenidamente. Describa sus problemas de forma clara y precisa. Muestrenos los mensajes de error o registros si los hay.

1.7. Contribuidores

El coordinador del proyecto actual es *Jan Willamowius* <http://www.willamowius.de/> <*jan@willamowius.de*>

Las principales caracteristicas y funciones de la version 2.0 lo son gracias a *Chih-Wei Huang* <http://www.cwhuang.idv.tw/> <*cwhuang@linux.org.tw*> y *Citron Network Inc.* <http://www.citron.com.tw/>, incluyendo el registro a traves de thread-safe y tablas de llamada, nueva arquitectura de modo de enrutado, proxy H.323, H.235 authentication and MySQL backend.

Michal Zygmuntowicz <*m.zygmuntowicz@onet.pl*> hizo un gran trabajo en el soporte de Radius y otras mejoras.

La version inicial del gatekeeper fue desarrollada por Xiang Ping Chen, Joe Metzger y Rajat Todi.

2. Compilacion e Instalacion

2.1. Compilando el Gatekeeper

Para crear el gatekeeper se necesitan al menos PWLib 1.5.0 y OpenH323 1.12.0 o superiores. La version en desarrollo del gatekeeper normalmente necesita la ultima version disponible de OpenH323. Esas librerias estan disponibles en la *página de descargas de OpenH323* <http://www.openh323.org/code.html>. Consulte las instrucciones sobre *como compilar el código OpenH323* <http://www.openh323.org/build.html>.

Orden de compilacion:

1. Librería PWLib (versiones release y/o debug)
2. Librería OpenH323
3. Pruebas a la aplicación OpenH323 (no es necesaria, solo para asegurarse de que todo funciona correctamente)
4. El Gatekeeper

En Unix haga un `configure` y `make debug` o `make opt` en el directorio del gatekeeper para crear la version debug o release, respectivamente. Use `make both` para crear ambas versiones. Recuerde que tiene que usar el GCC 3.3.x o superior. Las versiones anteriores pueden no funcionar. Una buena practica es hacer un `make debugdepend` o `make optdepend` en el directorio del gatekeeper antes de empezar la compilacion actual (make debug o make opt) - estos comandos crean las listas de dependencias apropiadas, asi despues de que actualice las fuentes desde el CVS, todos los ficheros afectados seran recompilados. De lo contrario puede acabar con el Gatekeeper compilado parcialmente con las antiguas y nuevas cabeceras - algo muy malo. En MS Windows simplemente abra y compile el proyecto facilidato (`gk.dsw`) para Microsoft Visual C++ 6.0 o 7.0 (Visual C++ 5.0 es demasiado antiguo).

Teclee `configure -help` para ver una lista detallada de todas las opciones de compilacion. Puede usarlas para activar o desactivar algunas caracteristicas del gatekeeper. Por ejemplo, si no necesita us RADIUS simplemente teclee: `configure -disable-radius`.

Para crear el gatekeeper con las librerias de OpenH323 y del sistema enlazadas estaticamente, tiene que usar `make optnoshared` o `make debugnoshared` .

Para poder usar el gatekeeper en entornos de gran carga, se recomienda activar la caracteristica LARGE_FDSET (configure –with-large-fdset=1024). La mayoria de los sistemas tambien necesitan recompilar/reconfigurar el kernel para poder permitir que mas de 1024 sockets sean asignados a un solo proceso. Recuerde que la libreria PWLib tambien empieza a soportar desde la version 1.5.3 una caracteristica similar, asi que puede elegir entre LARGE_FDSET GnuGk y la implementacion de PWLib. La implementacion nativa de GnuGk da mejor rendimiento. El valor maximo de LARGE_FDSET debe de ser calculado basandose en la prediccion del maximo numero de sockets usados. Una regla a ojo podria ser:

```
MAX_NUMBER_OF_CONCURRENT_CALLS * 10 * 120%

Donde:
10 = 2 sockets para Q.931 + 2 sockets para H.245 + 6 sockets para RTP y otras cosas
```

Asi para 100 llamadas simultaneas no necesita mas de 1024 sockets en el LARGE_FDSET.

2.2. Instalacion del Gatekeeper

No se necesita un procedimiento especial de instalacion. Simplemente copie el ejecutable en el directorio que usted quiera y cree un fichero de configuracion para este. Hay muchos ejemplos de configuracion en el subdirectorio `etc/` del arbol del codigo fuente. Consulte la seccion 4.2 (Fichero de Configuration) para una explicacion mas detallada.

Por ejemplo, en una plataforma Linux x86, el ejecutable `gnugk` optimizado es creado en el subdirectorio `obj_linux_x86_r/`. Puede copiarlo a `/usr/sbin/`, crear una configuracion en `/etc/gnugk.ini` y ejecutarlo mediante

```
$ /usr/sbin/gnugk -c /etc/gnugk.ini -o /var/log/gnugk.log -ttt
```

Consulte la seccion 4.1 (Opciones de linea de comandos) para más detalles.

2.3. Binarios pre-compilados

Si no desea compilar el gatekeeper desde las fuentes, hay muchos 'paquetes' pre-compilados disponibles desde *SourceForge* <http://sourceforge.net/project/showfiles.php?group_id=4797>. No todas las versiones seran puestas a disposicion como binarios - compruebe cual esta disponible.

Paquetes Tar (.tgz or .tar.gz)

Descarge el fichero tar y intruduzca el siguiente comando bajo `root`, sustituya el nombre del fichero por el del que ha descargado.

```
$ tar xvzf gnugk-x.x.x.tar
```

Paquetes Debian (.deb)

Debian incluye el gatekeeper como el paquete openh323gk. Puede instalarlo usando el siguiente comando bajo `root`:

```
$ apt-get install openh323gk
```

3. Empezando(Tutorial)

3.1. Un simple primer experimento

Para ver que todos los componentes estan cargados y funcionando, coja 2 estaciones de trabajo Linux, ambas conectadasd a la LAN. Asegurese de que tiene al menos la version 1.1 de OpenH323 y OhPhone instalados. En la primera maquina ejecute el gatekeeper y ohphone (en consolas diferentes:

```
jan@machine1 > gnugk -ttt
```

Ahora el gatekeeper está ejecutandose en modo directo. La opcion `"-ttt"le dice al gatekeeper que genere mucha salida para debug en la consola (puede dirigir esa salida a un fichero con "-o logfile").`

```
jan@machine1 > ohphone -l -a -u jan
```

Ahora este OhPhone está escuchando (-l) para recibir llamadas y las aceptara automaticamente (-a). Se ha registrado como usuario jan con el gatekeeper que detectara automaticamente. (si la autodeteccion fallara por alguna razon, use "-g 1.2.3.4"para especificar la IP donde esta escuchando el gatekeeper.)

En la segunda maquina ejecute solamente ohphone:

```
peter@machine2 > ohphone -u peter jan
```

La segunda instancia de OhPhone se registra con el gatekeeper auto detectado como usuario peter y intenta llamar al usuario jan. El gatekeeper resolvera el nombre de usuario a la IP desde donde el usuario jan se registro (maquina1 en este caso) y OhPhone llamara la otra instancia de OhPhone en la maquina uno.

La primera instancia de OhPhone aceptara esta llamada y Peter y Jan podran hablar.

3.2. Usando la interface de Status para monitorizar el gatekeeper

Ahora intente ver que mensajes han sido manejados por el gatekeeper. En la nueva consola en maquina1 vamos a usar telnet para conectar al gatekeeper:

```
jan@machine1 > telnet machine1 7000
```

Probablemente recibiremos un mensaje de acceso denegado (."ccess forbidden!"), porque no se le permite a todo el mundo espiar.

Ahora crearemos un fichero llamado gatekeeper.ini y lo colocaremos en el directorio donde arrancamos el gatekeeper. gatekeeper.ini solo contiene 4 lineas:

```
[Gatekeeper::Main]
Fortytwo=42
[GkStatus::Auth]
rule=allow
```

Pare el gatekeeper con Ctrl-C y ejecutelo de nuevo. Cuando hagamos el telnet de nuevo, permaneceremos conectados con el gatekeeper. Ahora repita el primer experimento donde Peter llama a Jan y observe que mensajes son manejados por el gatekeeper en el modo no-enrutado. En esta sesion de telnet pueden ser enviados algunos comandos: Teclee "help"para verlos. Para finalizar la sesion de telnet con el gatekeeper teclee "quitz presione Enter.

3.3. Arrancando el gatekeeper en modo enrutado.

Arrancando el gatekeeper en modo enrutado significa que el gatekeeper usa "señales de enrutado gatekeeper"para todas las llamadas. En este modo el gatekeeper todos los mensajes de señalicacion del gatekeeper pasaran a traves del gatekeeper y esto permite mucho mas control sobre las llamadas.

```
jan@machine1 > gnugk -r
```

Ahora el gatekeeper esta ejecutandose en modo enrutado. Haga telnet al puerto de estado y realice una llamada para ver que mensajes son manejados ahora por el gatekeeper.

Recuerde que todos los paquetes multimedia (audio y video) son todavia enviados directamente entre los endpoints (las 2 instancias de ohphone).

Desde que la señalicacion enrutada del gatekeeper es mucho mas complicada usted tiene mas posibilidades de encontrarse con un bug en el gatekeeper en este modo. Pero si se rompe, usted tiene que recoger las piezas. ;-)

3.4. Una centralita (PBX) virtual: desconectando llamadas

Hasta ahora el gatekeeper ha actuado solamente como un mecanismo para resolver nombres simbolicos a direcciones IP. Es una funcion importante pero dificilmente excitante.

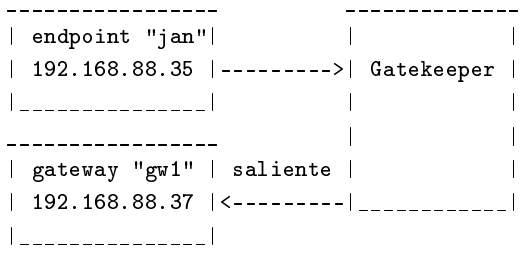
Desde que el gatekeeper tiene un gran control sobre las llamdas, este puede finalizarlas por ejemplo. Cuando estamos conectados al puerto de estado, podemos listar todas las llamadas activas con "PrintCurrentCalls". Para finalizar una llamada, podemos decir "Disconnectip 1.2.3.4"para uno de estos endpoints.

Uno puede por ejemplo escribir un script simple que conecte al puerto de estado y escuche todas las llamadas entrantes y las termine despues de 5 minutos, asi ningun usuario puede sobrepasar los recursos del sistema.

Eche un vistazo a las otra funciones de telefonia como TransferCall para ver que mas hay disponible.

3.5. Enrutando llamadas sobre un gateway para alcanzar usuarios externos

Sin usar un gateway usted solo puede llamar a otras personas con un telefono IP a traves de internet. Para alcanzar gente con telefonos ordinarios debe de usar un gateway.



El gatekeeper tiene que conocer que llamadas deben de ser supuestamente enrutadas a traves del gateway y que numeros deben ser llamados directamente. Use la seccion [RasSrv::GWPrefixes] del fichero de configuracion para decirle al gatekeeper el prefijo de numeros que deben de ser enrutados a traves del gateway.

```
[RasSrv::GWPrefixes]
gw1=0
```

Esta entrada le dira al gatekeeper que enrute todas las llamadas a numeros E.164 que empiezen por 0 al gateway que se ha registrado con el alias "gw1". Si no hay ningun gateway registrado con este alias la llamada fallara. (Recuerde que debe usar un aligas para el gateway - no puede simplemente decirle al gatekeeper la IP del gateway.)

Un prefijo puede contener los digitos 0-9, # y *. Tambien puede contener un caracter especial . (un punto) que sustituye cualquier digito y puede ser prefijado con ! (un

signo de exclamacion final) para desactivar el prefijo. La sustitucion de prefijos se hace de acuerdo a la regla de sustitucion de prefijos mas larga, teniendo las reglas ! mayor prioridad si la longitud es igual. Algunos ejemplos:

```
[RasSrv::GWPrefixes]
; Esta entrada enrutara numeros que empiecen con 0048 (pero no con 004850 y 004860)
; a gw1
gw1=0048,!004850,!004860
; Esta entrada sustituira solamente 001 y 10 digitos a continuacion
gw2=001.........
```

3.6. Reescribiendo numero E.164

Mientras este usando un gateway frecuentemente tendra que usar internamente numeros diferentes y reescribirlos antes de enviarlos a traves del gateway a una red telefonica. Puede usar la seccion 5.2 (RasSrv::RewriteE164) para configurar esto.

Ejemplo: Usted quiere llamar el numero 12345 con su telefono IP y le gustaria alcanzar el numero 08765 detras de un gateway llamado "gw1".

```
[RasSrv::GWPrefixes]
gw1=0

[RasSrv::RewriteE164]
12345=08765
```

Usted puede además configurar la reescritura de números E.164 basándose desde que gateway usted está recibiendo la llamada o enviando una llamada para que utilice la sección 5.3 (RasSrv::GWRewriteE164).

Ejemplo: Usted tiene dos gateways diferentes ("gw1z "gw2") hacia los cuales usted está enviando llamadas con prefijos 0044, pero los mismos requieren que sean agregado un prefijo diferente en el número después de que el ruteo ha elegido el gateway. Esto podría ser para propósitos de identificación por ejemplo.

```
[RasSrv::GWPrefixes]
gw1=0044
gw2=0044

[RasSrv::GWRewriteE164]
gw1=out=0044=77770044
gw2=out=0044=88880044
```

Ejemplo: Usted necesita identificar llamadas desde un gateway particular "gw1çon un prefijo específico antes que pasen estas llamadas hacia otro gateway "gw2".

```
[RasSrv::GWPrefixes]
gw2=1

[RasSrv::GWRewriteE164]
gw1=in=00=123400
```

La reescritura de expresiones acepta los caracteres comodín como '.' y signos de porcentaje '%' para permitir construír reglas más generales. El caracter punto puede estar en cuanquier lado de las expresiones (izquierda o derecha), el signo de porcentaje solo puede estar al la izquierda. Use '.' para corresponder cualquier caracter y copiar éste en la cadena reescrita y '%' para enparejar cualquier caracter e ignorar éste. Aqui uno ejemplos simples:

```
[RasSrv::RewriteE164]
; Reescribe 0044 + 7 dígitos mínimo hacia 44 + 7 dígitos mínimo
0044.......=44.......
; Reescribe números que empiezan con 11 + 4 dígitos + 11  hacia 22 + 4 dígitos + 22 (así 11333311 => 22333
11....11=22....22
; quita los primeros cuatro dígitos de todos los nñumeros (11114858345 => 4858345)
; esto es equivalente a 10 reglas %%%%1=1, %%%%2=2, ...
%%%%.=.
; inserta dos ceros en medio del núumero (111148581234 => 11110048581234)
....48=....0048
; incluso esto es posible (415161 => 041051061)
4.5.6=04.05.06
```

4. Configuración Básica del Gatekeeper

El funcionamiento del gatekeeper está completamente determinado por las opciones de la línea de comandos y el fichero de configuración. Algunas opciones de la línea de comandos pueden sustituir a opciones del fichero de configuración. Por ejemplo, la opción -l sustituye al atributo TimeToLive del fichero de configuración, prevaleciendo la introducida en la línea de comandos.

4.1. Opciones de la Línea de Comandos

Casi todas las opciones tienen una forma corta y una larga, por ejemplo, -c es lo mismo que -config.

4.1.1. Lo Básico

-h -help

Muestra todas las opciones disponibles y sale del programa.

-c -config filename

Especifica el fichero de configuración a usar.

-s -section section

Especifica cual sección principal usar en el fichero de configuración. Por defecto es [Gatekeeper::Main].

-i -interface IP

Especifica la interfaz (número IP) por el que el gatekeeper escuchará. Puedes dejar esta opción para permitir al gatekeeper determinar automáticamente la IP en la que escuchará, a menos que desees que el gatekeeper solo se enlace a una IP especificada.

-l -timetolive n

Especifica el tiempo de vida (en segundos) para el registro de los terminales. Este prevalece sobre el atributo TimeToLive del fichero de configuración. Ver 4.2.1 (aquí) para explicaciones detalladas.

-b -bandwidth n

Especifica el total de ancho de banda disponible para el gatekeeper. Si no se especifica esta opción, la administración de ancho de banda se desactiva por defecto.

-pid filename

Especifica el fichero para el pid, solo válido para versiones Unix.

-u -user name

Ejecuta el proceso del gatekeeper como este usuario. Solo válido para versiones Unix.

4.1.2. Modo del Gatekeeper

Las opciones de esta subsección prevalecen sobre los atributos de la 4.2.4 (sección [RoutedMode]) del fichero de configuración.

-d -direct

Usa marcado de llamada de terminal directo.

-r -routed

Usa marcado de llamada encaminado a través del gatekeeper.

-rr -h245routed

Usa marcado de llamada y canal de control H.245 encaminado a través del gatekeeper.

4.1.3. Información de Depuración

-o -output filename

Escribe anotaciones de trazado al fichero especificado.

-t -trace

Ver trazado verboso (verbosity). Cuantas más -t añadas, más verbosidad será lanzada a la salida. Por ejemplo, usa -ttttt para configurar el nivel de trazado al 5.

4.2. Archivo de Configuración

El archivo de configuración es un archivo estándar de texto. El formato es el siguiente:

```
[Section String]
Key Name=Value String
```

Los comentarios están marcados con un signo de numeral(#) o un punto y coma(;) al inicio de cada línea.

El archivo complete.ini contiene todas las secciones disponibles para el GnuGk. En la mayoría de los casos no tiene sentido utilizar todas las secciones al mismo tiempo. El archivo simplemente contiene una colección de ejemplos para muchas configuraciones.

El archivo de configuración puede ser cambiado en tiempo de ejecución (cuando el gnugk este ejecutándose). Una vez que haya modificado el archivo de configuración, usted puede editar el comando reload desde el puerto de estado (en una conexion telnet), o enviar una señal HUP al proceso del gatekeeper en ambientes UNIX. Por ejemplo,

```
kill -HUP `cat /var/run/gnugk.pid`
```

4.2.1. Sección [Gatekeeper::Main]

- Fortytwo=42
  Default: N/A Esta variable permite probar la existencia del archivo de configuración (gatekeeper.ini o cualquiera que sea su nombre) con el que va a trabajar el gatekeeper. Un mensaje de advertencia se mostrará en caso de que no existiera dicho archivo.

- Name=OpenH323GK
  Default: OpenH323GK Este es el identificador para el gatekeeper; es decir el nombre con que se lo conoce dentro del ambiente VOIP. Todos los mensajes enviados a los endpoints llevan este identificador.

- Home=192.168.1.1
  Default: 0.0.0.0 El gatekeeper escuchará por peticiones desde esta dirección IP. Por defecto, el gatekeeper escucha desde todas las interfaces de sus equipos. Usted puede quitar esta opción a menos que usted desee que el gatekeeper esté escuchando desde direcciones específicas. Múltiples direcciones pueden agregarse separadas por punto y coma (;) o coma (,).

- NetworkInterfaces=192.168.1.1/24,10.0.0.1/0
  Default: N/A Aquí se especifica las interfaces de red del gatekeeper. Por defecto el gatekeeper detectará las interfaces de sus equipos automáticamente. Existen dos situaciones en las que usted podría usar esta opción. Una es en el caso de que la detección automática fallara y la otra cuando el gatekeeper esta detrás de un NAT box y permite a endpoints con claves públicas registrarse con el gatekeeper.

- EndpointIDSuffix=_gk1
  Default: _endp El gatekeeper asignará un único identificador a cada endpoint registrado. Esta opción puede ser usada para especificar un sufijo y añadirlo al identificador del endpoint. Esto es muy utilizado solamente cuando se utiliza más de un gatekeeper.

- TimeToLive=300
  Default: -1 El registro del endpoint con el gatekeeper puede tener un límite en el tiempo de duración. El gatekeeper especifica la duración del registro de un endpoint incluyendo un campo timeToLive en el mensaje RCF. Después del tiempo especificado, el registro expira. Los endpoints enviarán periódicamente un mensaje RRQ con el bit keepAlive fijado antes del tiempo de expiración. Este tipo de mensajes incluyen un mínimo de información como esta descrito en H.225.0. Por eso es llamado como

lightweight RRQ. Esta configuración realizada especifica el cronometro time- to-live en segundos hasta que el registro expira. Debe tener muy en cuenta que el endpoint puede pedir un timeToLive muy corto al gatekeeper en el mensaje RRQ. Para evitar una sobrecarga de mensajes RRQ, el gatekeeper ajusta automáticamente este tiempo a 60 segundos si usted estableció un valor menor.

Después del tiempo de expiración, el gatekeeper enviará seguidamente dos mensajes IRQ para consultar si el endpoint esta aun levantado. Si el endpoint responde con un mensaje IRR, el registro se extenderá. En otro caso el gatekeeper enviará un mensaje URQ con razón ttlExpired hacia el endpoint. El endpoint debe entonces reintentar el registro con el gatekeeper usando un mensaje RRQ completo. Para deshabilitar esta opción, establezca como valor: - 1

- TotalBandwidth=100000
  Default: -1 Aquí se define el total del ancho de banda para ser dado a los endpoints. Por defecto esta característica esta deshabilitada. Tenga cuidado cuando use ésta, puesto que muchos endpoints tienen defectos de implementación.

- RedirectGK=Endpoints $>$ 100 $||$ Calls $>$ 50
  Default: N/A Esta opción le permite redireccionar endpoints hacia gatekeeper alternos cuando el gatekeeper se sobrecarga. Por ejemplo, en la configuración anterior, el gatekeeper rechazará un RRQ si los endpoints registrados sobrepasan los 100 o rechazará un ARQ si las llamadas actuales sobrepasan las 50. Además, usted puede redireccionar todos los endpoints de manera explícita configurando esta opción ya sea de manera temporal o permanente. El gatekeeper devolverá un mensaje de rechazo RAS con una lista de gatekeepers alternos definida en AlternateGks. Debe tener presente que un redireccionamiento permanente significa que los endpoints redireccionados no se registrarán nuevamente con este gatekeeper. Además tenga en cuenta que esta función solamente tiene efecto con el estándar H323 versión 4.

- AlternateGKs=1.2.3.4:1719:false:120:OpenH323GK
  Default: N/A Con esta opción nosotros permitimos la existencia de otro gatekeeper para proveer redundancia. Actualmente, usted podría encontrarse en una situación (válida por supuesto) donde algunos endpoints están registrados con el primer gatekeeper y algunos están registrados con el segundo. De igual manera usted debe ser capaz de usar los dos gatekeepers en un modo round robin para compartir la carga. Al hablar de gatekeeper primario se refiere al gatekeeper que estamos configurando actualmente y gatekeeper alterno significa el otro. El gatekeeper primario incluye un campo en el RFC que le indica al endpoint cual IP e identificación del gatekeeper alterno utilizar. Pero en cambio el gatekeeper alterno necesita conocer todos los registros realizados por el gatekeeper primario o sino este rechazará las llamadas. Por consiguiente nuestro gatekeeper puede remitir cada RRQ hacia una dirección IP alterna.

  La opción de configuración AlternateGKs especifica los campos contenidos en el RCF del gatekeeper primario . Los dos primeros campos del string definen donde (ip:port) se va a remitir la llamada. El tercer campo le indica a los endpoints si ellos necesitan registrarse con el gatekeeper alterno antes de de que tengan lugar las llamadas. Ellos normalmente no lo hacen puesto que nosotros remitimos sus mensajes RRQ, de esta manera ellos se registran con el gatekeeper alterno también. El cuarto campo especifica la prioridad para este gatekeeper. Si es pequeña, mejor, usualmente el gatekeeper primario es considerado para que tenga prioridad 1. Finalmente el último campo especifica el identificador del gatekeeper alterno.

- SendTo=1.2.3.4:1719
  Default: N/A A pesar de que esta información esta incluida en el gatekeeper alterno (AlternateGKs), you must still specify which address to forward RRQs to. Esto podría diferir de las direcciones del gatekeeper alterno, de esta manera esta es una opción de configuración aparte (piense en máquinas multihomed).

- SkipForwards=1.2.3.4,5.6.7.8
  Default: N/A Para evitar reenvíos circulares, usted no debería reenviar los RRQs que usted obtiene desde otros gatekeepers (Esta declaración es verdadera para ambos el gatekeeper primario y alterno). Dos mecanismos son utilizados para identificar si una petición debería ser reenviada. El primero busca por una bandera en los mensajes RRQ. Puesto que pocos endpoints implementan esto, nosotros necesitamos una segunda manera que sea más fiable. Especificar la IP del otro gatekeeper en esta lista.

- StatusPort=7000
  Default: 7000 Este es el puerto de estado para monitorear el gatekeeper. Ver 12 (this section) para más detalle.

- StatusTraceLevel=2
  Default: 2 Aquí se especifica el nivel de rastreo de mensajes para nuevas interfaces clientes. Ver 12 (this section) para más detalle.

- TimestampFormat=ISO8601
  Default: Cisco Esta opción permite controlar el formato de impresión de los strings generados por el gatekeeper. Esta opción afecta a las secciones: 8.4 ([SqlAcct]), 8.3 ([RadAcct]), 8.2 ([FileAcct]) y otras, excepto a 11.1 ([CallTable]). Además usted puede adaptar este formato de acuerdo a sus necesidades.

  Hay cuatro formatos predefinidos:

  - RFC822 - Este es el formato por defecto utilizado por el gatekeeper (ejemplo: Wed, 10 Nov 2004 16:02:01 +0100).
  - ISO8601 - Formato estándar ISO (ejemplo: 2004- 11- 10 T 16:02:01 +0100).
  - Cisco - Formato utilizado por equipos Cisco (ejemplo: 16:02:01.534 CET Wed Nov 10 2004).
  - MySQL - Formato simple que MySQL entiende perfectamente (ejemplo: 2004- 11- 10 16:02:01).

  Si usted necesita otro formato, usted puede construir sus propios formatos, utilizando reglas conocidas desde la función strftime de C. En general, el formato del string consiste de caracteres regulares y códigos formateados, precedidos por un signo de porcentaje (%). Por ejemplo: "%Y-%m-%d and percent%%"dará como resultado "2004- 11- 10 and percent%". Algunos códigos formateados comúnes son:

  - %a - Nombre abreviado del día de la semana
  - %A - Nombre completo del día de la semana
  - %b - Nombre abreviado del mes
  - %B - Nombre completo del mes
  - %d - Dia del mes como un número decimal
  - %H - Hora en el formato 24 horas
  - %I - Hora en el formato 12 horas
  - %m - Mes en número decimal

- %M - Minuto en número decimal
- %S - Segundo en número decimal
- %y - Año sin siglo
- %Y - Año con siglo
- %u - Microsegundos en número decimal (Esta es una extensión del gnugk)
- %z - Abreviación de la zona horaria (+0100)
- %Z - Nombre de la zona horaria
- %% - Signo de porcentaje

- EncryptAllPasswords=1
  Default: 0 Esta opción habilita la encriptación de todos los passwords en la configuración (SQL passwords, RADIUS passwords, [Password] passwords, [GkStatus::Auth] passwords). Si habilitamos esta opción, todos los passwords deben estar encriptados utilizando la aplicación addpasswd. De otra manera solo [Password] y [GkStatus::Auth] passwords son encriptados (antiguo comportamiento).
- KeyFilled=0
  Default: 0 Define un byte de relleno global para ser utilizado durante la encriptación/ desencriptación de passwords. Usualmente, usted no necesitará cambiar esta opción.

La mayoría de los usuarios nunca necesitará cambiar cualquiera de los siguientes valores: Estos son utilizados para operaciones de testing o aplicaciones avanzadas.

- UseBroadcastListener=0
  Default: 1 Define si se escucha las peticiones broadcast RAS. Esto requiere enlazarse hacia todas las interfaces en una máquina, de esta manera si usted desea ejecutar múltiples instancias de gatekeepers sobre la misma máquina, usted debe deshabilitar esta opción.
- UnicastRasPort=1719
  Default: 1719 El identificador del canal RAS TSAP para unicast.
- MulticastPort=1718
  Default: 1718 El identificador del canal RAS TSAP para multicast.
- MulticastGroup=224.0.1.41
  Default: 224.0.1.41 El grupo multicast para el canal RAS.
- EndpointSignalPort=1720
  Default: 1720 Puerto por defecto para el canal de señalización de llamadas de endpoints.
- ListenQueueLength=1024
  Default: 1024 Longitud de la cola para conexiones TCP entrantes.
- SignalReadTimeout=1000
  Default: 1000 Tiempo en milisegundos para leer la interrupción en canales de señalización de llamadas (Q931).
- StatusReadTimeout=3000
  Default: 3000 Tiempo en milisegundos para leer la interrupción en canales de estado (status channel).

- StatusWriteTimeout=5000
  Default: 5000 Tiempo en milisegundos para escribir la interrupción en canales de estado (status channel).

4.2.2. Sección [GkStatus::Auth]

En esta sección se definen un número de reglas para determinar quienes están permitidos de conectarse al gatekeeper vía puerto de estado (vía telnet). Quien quiera que tenga acceso al puerto de estado tiene un control completo sobre el gatekeeper. Asegúrese de que ésta sección esté configurada correctamente.

- rule=allow
  Default: forbid Posibles valores son:
  - forbid - Prohíbe cualquier conexión.
  - allow - Permite cualquier conexión.
  - explicit - Lee el parámetro ip=value donde ip es la dirección IP del cliente observado, value puede ser 1,0 o allow,forbid o yes,no. Si ip no está en la lista el parámetro default es usado.
  - regex - La IP del cliente se hace corresponder con la expression regular dada.

    Ejemplo:

    Para permitir clientes desde 195.71.129.0/24 y 195.71.131.0/24:

    regex=^195\.71\.(129|131)\.[0-9]+$
  - password - El usuario tiene un username y password apropiado para conectarse (login). El formato de username/password es el mismo que el de la sección 7.2 ([SimplePasswordAuth]).

  Por otra parte, estas reglas puede ser combinadas por "|.º "&". Por ejemplo,
  - rule=explicit | regex
    La IP del cliente debe validarse con las reglas explicit o regex.
  - rule=regex & password
    La IP del cliente debe validarse con las reglas regex, y el usuario tiene que conectarse con un username y password.
- default=allow
  Default: forbid Esta opción es utilizada solamente cuando rule=explicit.
- Shutdown=forbid
  Default: allow Esta opción es utilizada para especificar si se permite o no apagar el gatekeeper vía puerto de estado.
- DelayReject=5
  Default: 0 Esta opción especifica que tiempo (en segundos) se debe esperar antes de rechazar un username / pa s sword inválido.

4.2.3. Sección [LogFile]

En esta sección se define parámetros relacionados con el archivo de log. Actualmente se permite a los usuarios especificar las opciones de rotación.

- Rotate=Hourly | Daily | Weekly | Monthly
  Default: N/A Si se utiliza esta opción. El archivo de log rotará basado en esta configuración. La rotación Hourly habilita la rotación una vez por hora, la rotación Daily, una vez por día, la Weekly, una vez por semana y la Monthly, una vez por mes. Para definir una rotación exacta ésta esta determinada por una combinación de las variables RotateDay y RotateTime. Durante la rotación, un archivo existente es renombrado con el formato CURRENT_FILENAME.YYYYMMDD-HHMMSS, donde YYYYMMDD-HHMMSS es reemplazada con la fecha actual, y nuevas líneas son agregadas a un archivo vacío. Para deshabilitar la rotación, no utilice la opción Rotate o establézcala a 0.

  Ejemplo1: Rotación cada hora (00:45, 01:45, ..., 23:45):

  ```
  [LogFile]
  Rotate=Hourly
  RotateTime=45
  ```

  Ejemplo 2: Rotación cada día a las 23:00 (11PM):

  ```
  [LogFile]
  Rotate=Daily
  RotateTime=23:00
  ```

  Ejemplo 3: Rotación cada Domingo a las 00:59:

  ```
  [LogFile]
  Rotate=Weekly
  RotateDay=Sun
  RotateTime=00:59
  ```

  Ejemplo 4: Rotación en el último día de cada mes:

  ```
  [LogFile]
  Rotate=Monthly
  RotateDay=31
  RotateTime=23:00
  ```

4.2.4. Sección [RoutedMode]

Los mensajes de señalización de llamada pueden ser manejados de dos maneras. El primer método es Señalización de llamada en modo directo (Direct Endpoint Call Signalling), en el cual los mensajes de señalización de llamada son intercambiados directamente entre los endpoints. El segundo método es Señalización de llamada mediante gatekeeper (Gatekeeper Routed Call Signalling). En este método, los mensajes de señalización de llamada son enrutados a través del gatekeeper entre los endpoints. La selección de cual método utilizar es realizada por el gatekeeper.

Cuando se utiliza la señalización de llamadas en modo gatekeeper (Gatekeeper Routed Call Signalling), el gatekeeper puede seleccionar si enrutar o no el canal de control H.245 (H.245 control channel) y los canales lógicos (logical channels).

Caso I.

El gatekeeper no rutea estos canales. El canal de control H.245 (H.245 control channel) y los canales lógicos (logical channels) se establecen directamente entre los endpoints.

Caso II.

El canal de control H.245 (H.245 control channel) se enruta entre los endpoints a través del gatekeeper, mientras que los canales lógicos (logical channels) se establecen directamente entre los endpoints.

Caso III.

El gatekeeper rutea el canal de control H.245 (H.245 control channel), asi como también los canales lógicos (logical channels), incluyendo el RTP/RTCP para audio y video, y el canal T.120 para datos. En este caso, ningún tipo de tráfico es intercambiado directamente entre los endpoints. Esto es usualmente llamado un Proxy H.323, el mismo que puede ser considerado también como un gateway H.323-H.323.

Esta sección define las opciones de ruteo del modo gatekeeper (Gatekeeper Routed) (Casos I & II). La característica de proxy se definirá más adelante en la 4.2.5 (siguiente sección). Todas las configuracones dentro de esta sección son afectadas por el comando reload desde el puerto de estado.

- GKRouted=1
  Default: 0 Definir si se habilita o no el ruteo en modo gatekeeper (gatekeeper routed mode).
- H245Routed=1
  Default: 0 Definir si se rutea también el canal de control H.245 (H.245 control channel). Solamente tendrá efecto si el parámetro GKRouted=1.
- CallSignalPort=0
  Default: 1721 El puerto por donde el gatekeeper realizará la señalización de llamada. El puerto por defecto es 1721. Nosotros no usamos el puerto conocido 1720 de esta manera usted podría ejecutar un endpoint H.323 en la misma máquina del gatekeeper. Usted puede establecer este parámetro en 0 para permitirle al gatekeeper seleccionar un puerto arbitrario.
- CallSignalHandlerNumber=2
  Default: 1 El numero de manejadores de señalización de llamadas. Usted puede incrementar este número en un gatekeeper fuertemente cargado. Este número puede ser incrementado en tiempo de ejecución del gatekeper. Si usted tiene una maquina SMP, usted puede establecer este número al número de CPUs que tenga.
- RtpHandlerNumber=2
  Default: 1 El número de hilos manejadores de RTP proxy.
- AcceptNeighborsCalls=1
  Default: 1 Con esta característica habilitada, el hilo de señalización de llamada aceptará llamadas sin un CallRec preexistente dentro del CallTable, de tal manera que un endpoint correspondiente a una dirección de destino (destinationAddress) en un mensaje Setup pueda ser encontrado en la tabla de registro (RegistrationTable), y el emisor de la llamada es su vecino o su GK padre. El gatekeeper además utiliza su propia dirección de señalización de llamada (call signalling address) dentro de LCF en respuesta a un LRQ. Eso significa, la señalización de llamada será ruteada

hacia el GK2 en llamadas GK-GK. Como resultado, los CDRs en el GK2 pueden mostrar correctamente el tiempo de conexión en lugar de 'unconnected'.

- AcceptUnregisteredCalls=1
  Default: 0 Con esta característica habilitada, el gatekeeper aceptará llamadas desde cualquier endpoint no registrado. Sin embargo, esto permite riesgos en la seguridad. Tenga cuidado con esto.

- RemoveH245AddressOnTunneling=1
  Default: 0 Algunos endpoints envían h245Address dentro del UUIE del Q.931 aún cuando h245Tunneling está establecida en TRUE. Esto podría causar problemas de interoperabilidad. Si la opción es TRUE, el gatekeeper removerá las h245Address cuando el indicador h245Tunneling esté en TRUE. Esto forzará a que el que está llamando permanezca en modo tunnelling.

- RemoveCallOnDRQ=0
  Default: 1 Con esta opción establecida en off, el gatekeeper no desconectará una llamada si éste recive un DRQ para la llamada. Esto evita que un DRQ alcance un Release Complete. Esto tiene significado solamente en modod de ruteo (routed mode) puesto que en modo directo (direct mode), el mecanismo de señal end-of-call es un DRQ.

- DropCallsByReleaseComplete=1
  Default: 0 De acuerdo con la recomendación H.323, el gatekeeper podrá colgar una llamada enviando un mensaje RAS DisengageRequest hacia los endpoints. Sin embargo, algunos endpoints mal implementados ignoran este comando. Con esta opción establecida en on, el gatekeeper enviará mensajes Q.931 Release Complete en lugar de mensajes RAS DRQ hacia ambos endpoints para forzar a que ellos terminen la llamada.

- SendReleaseCompleteOnDRQ=1
  Default: 0 Cuando se cuelga la llamada, el endpoint envía tanto un Release Complete dentro de H.225/Q.931 y un DRQ dentro de los mensajes RAS. Podría ocurrir que el DRQ sea procesado primero, ocasionando que el gatekeeper cierre el canal de señalización de llamadas, previniendo así que el Release Complete sea enviado hacia el otro endpoint. Aunque el gatekeeper cierre el canal TCP hacia el destino, algunos endpoints (por ejemplo Cisco CallManager) no terminan la llamada aún si el canal de señalización de llamada está cerrado. Esto ocasiona que teléfonos se mantengan sonando si el que llama cuelga antes de que el receptor conteste. Establecer este parámetro en 1 produce que el gatekeeper siempre envíe un Release Complete hacia ambos endpoints antes de que cierren la llamada cuando éste recibe un DRQ desde una de la partes.

- SupportNATedEndpoints=1
  Default: 0 Define si se permite a un endpoint detrás de una caja NAT registrarse con el gatekeeper. Si ésto se permite, el gatekeeper traducirá las direcciones IP dentro del canal Q.931 and H.245 hacia la IP de la caja NAT.

  Desde la versión 2.0.2, El GnuGk soporta salida de llamadas (NAT outbound calls) (desde un endpoint detrás de la NAT hacia redes públicas) directamente sin ninguna modificación necesaria de endpoints o de caja NAT. Solamente registre el endpoint con el GnuGk y usted puede hacer sus llamadas ahora.

- ScreenDisplayIE=MyID
  Default: N/A Modifica el DisplayIE del Q.931 hacia un valor específico.

- ScreenCallingPartyNumberIE=0965123456
  Default: N/A Modifica el CallingPartyNumberIE del Q.931 hacia un valor específico.
- ScreenSourceAddress=MyID
  Default: N/A Modifica el campo sourceAddress del elemento UUIE del mensaje Q.931 Setup.
- ForwardOnFacility=1
  Default: 0 Si se habilita, en recibir Q.931 Facility con razón callForwarded, el gatekeeper adelantará el call Setup directamente hacia el endpoint remitido, en lugar de pasar el mensaje de regreso al que realiza la llamada (caller). Si usted tiene endpoints con defectos de implementación que no pueden manejar Q.931 Facility con razón callForwarded, active esta opción. Tenga en cuenta que esta característica no siempre podría trabajar correctamente, puesto que ésta no proporciona ningun recurso de renegociación de capacidades y media channel reopening.
- ShowForwarderNumber=0
  Default: 0 Defina esta opción si desea reescribir el número del calling party al número del forwarder. Esta característica es utilizada para propósitos de billing. Solamente es válida si ForwardOnFacility=1.
- Q931PortRange=20000-20999
  Default: N/A (permitir que el SO asigne los puertos) Especifique el rango de puertos TCP para los canales de señalización Q.931. Tenga en cuenta que el tamaño del rango podría limitar al número de llamadas concurrentes.
- H245PortRange=30000-30999
  Default: N/A (permitir que el SO asigne los puertos) Especifique el rango de puertos TCP para los canales de control H.245. Tenga en cuenta que el tamaño del rango podría limitar el número de llamadas concurrentes.
- ConnectTimeout=60000
  Default: 180000 Tiempo de descanso en milisegundos para esperar antes de remover llamadas no conectadas (unconnected calls) de la tabla de llamadas (call table). Este es un cronometro de resguardo que no permite que las llamadas no conectadas (unconnected calls) se mantengan por siempre en la tabla de llamadas (call table). Tenga en cuenta que establecer este valor demasiado corto podría causar que las llamadas sean removidas antes de ser contestadas.
- TcpKeepAlive=0
  Default: 1 Habilite/Deshabilite la característica keepalive en sockets de señalización (TCP signaling sockets). Esto puede ayudar a detectar canales de señalización inactivos y prevenir llamadas inactivas (dead calls) imposibles de colgar en la tabla de llamadas (call table). Para que esta opción funcione, usted necesita además modificar las configuraciones del sistema para ajustar el "keep alive timeout". Para más detalle vea docs/keepalive.txt.

4.2.5. Sección [Proxy]

Esta sección define las características del proxy H.323. Esto quiere decir que el gatekeeper ruteará todo el tráfico entre el endpoint emisor y receptor (calling and called endpoints), de esta manera ho existe tráfico directamente entre dos endpoints. Por eso esto es muy utilizado si usted tiene algunos endpoints utilizando IP privadas detrás de una caja NAT y algunos endpoints utilizando IP publicas fuera de la caja.

El gatekeeper puede hacer proxy para canales lógicos de RTP/RTCP (audio y video) y T.120 (datos). Canales abiertos por procedimientos de conexión rápida (fast-connect) o H.245 tunnelling también son soportados.

Tenga en cuenta que para hacer que el proxy funcione, el gatekeeper debe tener conexión directa hacia ambas redes: la de emisor (caller) y la del receptor (callee).

- Enable=1
  Default: 0 Define si se habilita o no la función proxy. Usted debe de habilitar el ruteo en modo gatekeeper primero (ver la 4.2.4 (sección previa)). Usted no tiene que especificar el ruteo H.245. Este será automáticamente utilizado si se requiere.

- InternalNetwork=10.0.1.0/24
  Default: N/A Define las redes detrás del proxy. Se permite el uso de múltiples redes internas. El proxy rutea canales solamente de las comunicaciones entre un endpoint en la red interna y una externa. Si usted no especifica esto, todas las llamadas serán llevadas a través del proxy.

  Formato:

  InternalNetwork=network address/netmask[,network address/netmask,...]

  La netmask puede ser expresada en notación decimal punteada o en notación CIDR (prefix length), como se muestra en el ejemplo.

  Ejemplo:

  InternalNetwork=10.0.0.0/255.0.0.0,192.168.0.0/24

- T120PortRange=40000-40999
  Default: N/A (permitir que el SO asigne los puertos) Especifica el rango de números de puertos TCP para el canal de datos T.120. Tenga en cuenta que el tamaño del rango podría limitar el número de llamadas concurrentes.

- RTPPortRange=50000-59999
  Default: 1024-65535 Especifica el rango de números de puertos para los canales RTP/RTCP. Tenga en cuenta que el tamaño del rango podría limitar el número de llamadas concurrentes.

- ProxyForNAT=1
  Default: 1 Si se activa, el gatekeeper pasará por el proxy aquellas llamadas en las cuales uno de los endpoints participantes esta detrás de una caja NAT (NAT box). Esto asegura que el stream RTP/RTCP pueda penetrar dentro de la NAT box sin modifica este. Sin embargo, el endpoint que se encuentra detrás de la NAT box debe utilizar el mismo puerto para enviar y recibir el stream RTP/RTCP. Si usted tiene trabaja con endpoint con fallas o que no satisfacen la precondición, usted debe mejor deshabilitar esta característica y permitir que la NAT box remita el stream RTP/RTCP por usted.

- ProxyForSameNAT=0
  Default: 0 DEDefine si se habilita el proxy para llamadas entre endpoints desde la misma NAT box. Uste no necesita habilitar generalmente esta característica , ya que normalmente los endpoints de la misma NAT box pueden comunicarse entre si.

5. Configuración de Enrutado

Las siguientes secciones del archivo de configuración pueden ser usadas para configurar el enrutamiento de las llamadas.

5.1. Sección [Políticas de enrutado]

Esta sección explica el funcionamiento de las diversas políticas de enrutado del gatekeeper.

Las peticiones de llamada entrante pueden ser encaminadas usando un número de proovedores de enrutado:

- explicit
  El destinatario se detalla de manera explícita en la petición de enrutado.
- internal
  La regla clásica; buscar el destinatario en la tabla de registro RegistrationTable
- parent
  Enruta la llamada con la información obtenida de la respuesta del GK padre a un ARQ enviado por el gatekeeper.
- neighbor
  Enruta la llamada usando vecinos a través de mensajes LRQ
- dns
  El destino se obtiene del DNS, dado que es obtenible.
- vqueue

  Usar el mecanismo de cola virtual y generar un evento RouteRequest para permitir manejar el enrutado a una aplicación externa (sólo puede ser usada OnARQ)

Las políticas se prueban de manera sucesiva hasta hallar la correcta.

Estas políticas pueden ser aplicadas a diversos tipos de peticiones de enrutado, y datos de entrada de enrutado. Éstos son los siguientes: ARQ, LRQ, Setup y Facility (con la razón callForwarded) Existe también una política general, que viene a ser la opción por defecto frente a los otros tipos.

Example:

```
   [RoutingPolicy]
h323_ID=dns,internal
002=neighbor,internal
Default=internal,neighbor,parent
```

Cuando un mensaje que llama a una decisión de enrutado se recibe, todas las llamadas a un alias del tipo h323_ID serán resueltas a través de DNS. Si esto no soluciona el alias, se comprueba contra las tablas de registro internas. Si se pide un alias que comience con 002, los vecinos se comprueban antes de las tablas de registro. Si el alias solicitado no es del tipo h323_ID o comienza por 002, se usa la política por defecto consultando las tablas de registro internas, tras ellas, a los vecinos, y, en caso de fallo, al padre.

Para los mensajes ARQ, LRQ, Setup y Facility se deberían usar las secciones [RoutingPolicy::OnARQ], [RoutingPolicy::OnLRQ], [RoutingPolicy::OnSetup] y [RoutingPolicy::OnFacility] usando la sintaxis anteriormente explicada.

5.2. Sección [RasSrv::RewriteE164]

Esta sección define las reglas de reescritura para dialedDigits (números E.164)

Format:

[!]original-prefix=target-prefix

Si el número comienza con original-prefix, se reescribe a target-prefix. Si la bandera '!' precede a original-prefix, se invierte el sentido y el prefijo del objetivo se antepone al número marcado. Se permiten los comodines especiales ('.' y '%').

Example:

08=18888

Si se marca 08345718, se reescribe a 18888345718.

Example:

!08=18888

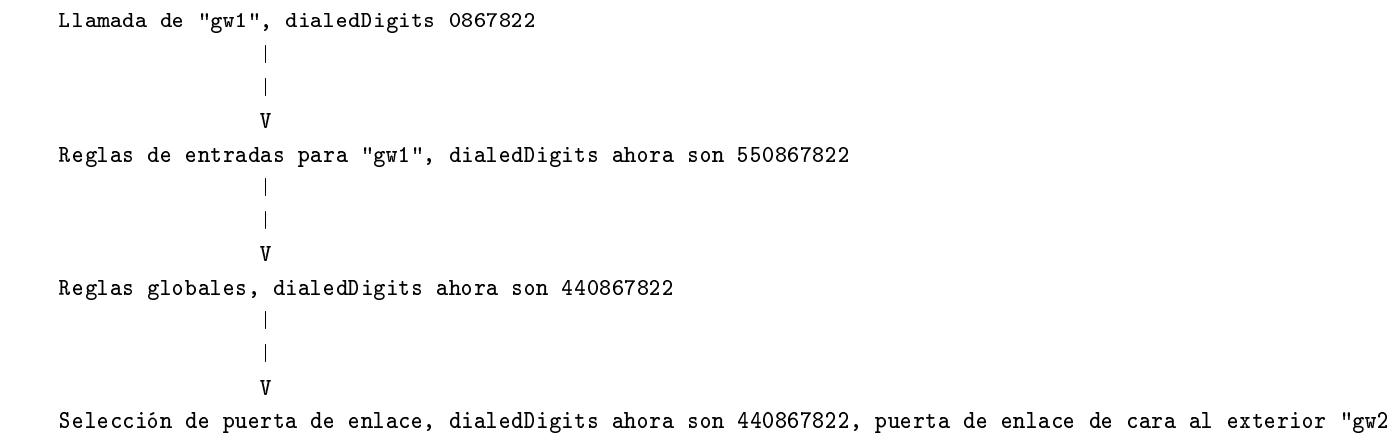
Si se marca 09345718, se reescribe a 1888809345718.

Opción:

- Fastmatch=08
  Por defecto: N/A Sólo reescribe dialDigits que comiencen con el prefijo especificado.

5.3. Sección [RasSrv::GWRewriteE164]

Esta sección describe la reescritura de números dialedDigits E.164 en función de si si llega o se envía la llamada de la puerta de enlace. Esto permite manipulación más flexible de los dialedDigits para enrutado y demás. En combinación con el 5.2 (RasSrv::RewriteE164) se puede reescribir en tres fases:


Llamada de "gw1", dialedDigits 0867822
Reglas de entradas para "gw1", dialedDigits ahora son 550867822
Reglas globales, dialedDigits ahora son 440867822
Selección de puerta de enlace, dialedDigits ahora son 440867822, puerta de enlace de cara al exterior "gw2

```
                |
                V
Reglas de salida para "gw2", dialedDigits ahora son 0867822
                |
                |
                V
Llamada a "gw2", dialedDigits 0867822
```

Format:

gw-alias=in|out=[!]original-prefix=target-prefix[;in|out...]

Si la llamada encaja con la puerta de enlace, la dirección y comienza con original-prefix se reescribe a target-prefix. Si la bandera '!' precede a original-prefix, el sentido se invierte. Se permiten los caracteres especiales ('.' and '%'). Es preciso separar con ';' las diversas reglas para una misma puerta de enlace.

Example:

gw1=in=123=321

Si se recibe una llamada de "gw1.ª 12377897, se reescribe a 32177897 antes de acometer cualquier otra acción.

5.4. Sección [Endpoint::RewriteE164]

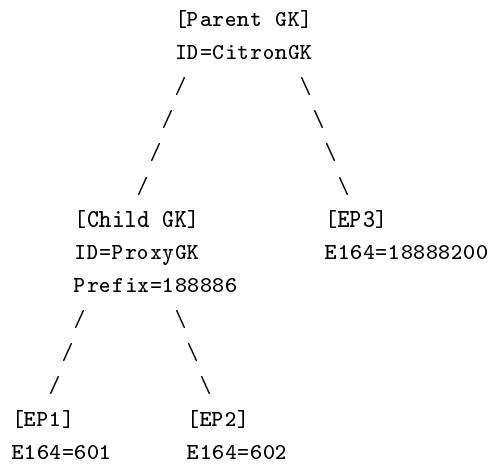
Una vez especificado el/los registro/s para el punto final de la puerta de enlace, el gatekeeper padre enrutará las llamadas con dialedDigits comenzando con esos prefijos.

El gatekeeper hijo puede reescribir el destino de acuerdo a las reglas especificadas en esta sección. En contraste, cuando un punto final interno llama a un punto final registrado en el padre, la fuente se escribirá a inversas.

Format:

external prefix=internal prefix

Por ejemplo, con la siguiente configuración:



Con la regla:

```
188886=6
```

Cuando EP1 llame EP3 por 18888200, el CallingPartyNumber en el Setup Q.931 se reescribirá a 18888601. A inversas, EP3 alcanza EP1 y EP2 llamando a 18888601 y 18888602, respectivamente. Por tanto, un punto final registrado al GK hijo con prefijo '6' aparecerá como un punto final con prefijo '188886', para los puntos finales registrados al padre.

La sección no se relaciona con la sección 5.2 (RasSrv::RewriteE164), aunque la última tendrá efecto antes.

6. Configuración RAS

6.1. Sección [RasSrv::GWPrefixes]

Esta sección lista qué números E.164 son ruteados hacia un gateway específico.

Formato:

```
gw-alias=prefix[,prefix,...]
```

Tenga en cuenta que usted tiene que especificar el alias del gateway. Si un gateway está registrado con el alias, todos los números que empiezan con los prefijos son ruteados hacia este gateway. Loas caracteres especiales . y ! puede ser usuada aqui para hacer corresponder cualquier dígito y deshabilitar el prefijo.

Ejemplo:

```
test-gw=02,03
```

6.2. Sección [RasSrv::PermanentEndpoints]

En esta sección usted puede poner endpoints que no tienen soporte RAS o que usted desea que su registro no expire. Los registros siempre se mantendrán en la tabla de registro del gatekeeper. Sin embargo, usted puede tambien des-registrar éste, mediante el puerto de estado. Los caracteres especiales . y ! pueden ser utilizados con prefijos aqui para hacer corresponder cualquier dígito y deshabilitar el prefijo.

Formato:

```
IP[:port]=alias[,alias,...;prefix,prefix,...]
```

Ejemplo:

Para gateway,

```
10.0.1.5=Citron;009,008
```

Para terminal,

```
10.0.1.10:1720=700
```

6.3. Sección [RasSrv::RRQFeatures]

- AcceptEndpointIdentifier=1
  Default: 1 Definir si aceptar o no endpointIdentifier que está especificado en un RRQ completo.

- AcceptGatewayPrefixes=1
  Default: 1 Un gateway puede registrar sus prefijos con el gatekeeper incluyendo supportedPrefixes dentro del campo terminalType del RRQ. Esta opción define si se acepta los prefijos específicos de un gateway.

- OverwriteEPOnSameAddress=1
  Default: 0 En algunas redes una dirección ip del endpoint puede cambiar inexplicablemente. Esto puede pasar cuando un endpoint está usando conexión PPP (ejemplo: modem or ADSL). Esta opción define cómo manejar una petición de registro (RRQ) desde una dirección IP la misma que no se corresponde con la que nosotros tenemos almacenada. La acción por default es rechazar la petición. Con esta opción habilitada la petición en conflicto provocaría una petición de des-registro (URQ) para que la dirección IP existente y la entrada sean removidas permitiendo que el endpoint se registre con su nueva dirección.

6.4. Sección [RasSrv::ARQFeatures]

- ArjReasonRouteCallToSCN=0
  Default: 1 Si es ¿es"(si), el gatekeeper rechaza una llamada que viene desde un gateway hacia si mismo por razones de routeCallToSCN.

- ArjReasonRouteCallToGatekeeper=1
  Default: 1 Si es ¿es", el gatekeeper rechaza un ARQ respondido sin que haya un pre-existente CallRec encontrado en la CallTable por causa de routeCallToGatekeeper en modo ruteo. El endpoint liberará la llamada imediátamente y reenviará un Call Setup hacia el gatekeeper.

- CallUnregisteredEndpoints=0
  Default: 1 Con esta opción habilitada, el gatekeeper aceptará un ARQ de un endpoint registrado con destCallSignalAddress, sin importar si la dirección le corresponde a un endpoint registrado o no. Eso significa que usted puede explícitamente especificar la IP del endpoint (registrado o no) que usted desee llamar.

- RemoveTrailingChar=#
  Default: N/A Especifique el caracter de rastreo a ser removido en destinationInfo. Por ejemplo, si su endpoint contiene incorrectamente el caracter de terminación como '#' dentro destinationInfo, usted puede remover éste mediante esta opción.

- RoundRobinGateways=0
  Default: 1 Habilite/Deshabilite la selección de un gateway mediante round-robin, si es que más de un gateway se corresponden con un número marcado. Si se deshabilita, el primer gateway disponible será seleccionado. De otra manera, las llamadas subsiguientes serán enviadas hacia los gateways utilizando el modo round-robin.

6.5. Section [NATedEndpoints]

El gatekeeper puede automáticamente detectar si un endpoint está detrás de una NAT. Sin embargo, si la detección falla, usted puede especificar éste manualmente en esta sección.

Formato:

alias=true,yes,1,...

Example:

Especifique un endpoint con alias 601 está detrás de la NAT.

```
601=true
```

7. Configuración de la Autenticación

Las siguientes secciones en el archivo de configuración pueden ser utilizadas para configurar la Autentificación.

7.1. Sección [Gatekeeper::Auth]

Esta sección define los mecanismos de autenticación para el gatekeeper.

Sintaxis:

```
authrule=actions

<authrule> := SimplePasswordAuth | AliasAuth | PrefixAuth | RadAuth | RadAliasAuth | ...
<actions>  := <control>[;<ras>|<q931>,<ras>|<q931>,...]
<control>  := optional | required | sufficient
<ras>      := GRQ | RRQ | URQ | ARQ | BRQ | DRQ | LRQ | IRQ
<q931>     := Setup | SetupUnreg
```

Una regla puede resultar en uno de los siguientes tres códigos: ok, fail, pass.

- ok - La petición es autenticada por este módulo.
- fail - La autenticación falla y debe ser rechazada.
- next - La regla no puede determinar la petición.

También hay tres maneras de controlar una regla:

- optional - Si la regla no puede determinar la petición, ésta es enviada a la siguiente regla.
- required - La petición debe ser autenticada por este módulo, de lo contrario ésta sería rechazada. La petición autenticada entonces sería enviada hacia la siguiente regla.
- sufficient - Si la petición es autenticada, ésta es aceptada, de lo contrario ésta sería rechazada. Es decir, la regla determina la suerte de la petición. Ninguna regla debe ser ubicada después de una regla sufficient, puesto que no tendrá ningún efecto.

Módulos soportados actualmente:

- SimplePasswordAuth/SQLPasswordAuth Estos módulos revisan los campos tokens or cryptoTokens de un mensaje RAS. Los tokens deben contener al menos un generalID y un password. Para aquellos tokens cryptoTokens, cryptoEPPwdHash separados (hashed) por MD5 simple y aquellos tokens nestedcryptoToken separados (hashed) por HMAC-SHA1-96 (libssl debe ser instalada) hay soporte actualmente. Para tokens tokens separados (hashed) por CAT (Cisco Access Token) y un username/password en texto claro son soportados actualmente. El ID y password son leídos desde la sección 7.2 ([SimplePasswordAuth]), y desde una base de datos SQL para los módulos SimplePasswordAuth y SQLPasswordAuth. El módulo MySQLPasswordAuth también es soportado.
- AliasAuth/SQLAliasAuth Este módulo puede ser utilizado solamente para auntenticar RegistrationRequest (RRQ). La dirección IP de un endpoint con un alias dado deben corresponderse con un patrón específico. Para AliasAuth el patrón está definido en la sección 7.4 ([RasSrv::RRQAuth]). Para SQLAliasAuth, el patrón es recuperado desde una base de datos SQL, que está definida en la sección 7.5 ([SQLAliasAuth]).
- PrefixAuth La dirección IP o alias de una petición con un prefijo dado deben corresponderse o emparejarse con un patrón específico. Ver la sección 7.6 ([PrefixAuth]) para más detalle. Actualmente el módulo puede autorizar solamente AdmissionRequest (ARQ) y LocationRequest (LRQ).
- RadAuth Provee autenticación basada en el esquema de seguridad H.235 username/password. Autentica RRQ, ARQ y Q.931 Setup mediante servidores RADIUS remotos. Envía al Servidor RADIUS los usernames y passwords extraídos desde los CAT (Cisco Access Tokens) tokens que viajan dentro de los paquetes RRQ, ARQ o Setup. Por consiguiente si sus endpoints no soportan CATs o usted no necesita el esquema de autenticación basado en usernames/password asignados individualmente - este módulo no le servirá (pero puede chequear además el módulo RadAliasAuth). Ver la sección 7.7 ([RadAuth]) para más detalle.
- RadAliasAuth Provee autenticación basada en alias de endpoints y/o direcciones IP con servidores remotos RADIUS. Esto no necesita ningún tokens H.235 dentro de los mensajes RAS, de esta manera esto puede ser utilizado sobre una amplia variedad de sistemas de igual manera como RadAuth. Los mensajes RRQ, ARQ y Q.931 Setup pueden ser autentificados utilizando este modulo. Ver la sección 7.8 ([RadAliasAuth]) para más detalle.

Usted puede además configurar una regla para chequear solamente algunos mensajes RAS particulares. El siguiente ejemplo configura SimplePasswordAuth como regla opcional para chequear mensajes RRQ y ARQ. Si un mensaje RRQ no es chequeado (no contiene campos tokens o cryptoTokens), éste es chequeado por AliasAuth. La regla por defecto es aceptar todas las peticiones.

Ejemplo 1:

```
SimplePasswordAuth=optional;RRQ,ARQ
AliasAuth=sufficient;RRQ
```

El ejemplo mostrado más abajo autentica todas las llamadas, chequeando los mensajes de señalización Setup de una manera detallada, utilizando el módulo RadAliasAuth.

Ejemplo 2:

```
RadAliasAuth=required;Setup
default=allow
```

Este ejemplo chequea los mensajes de registro de endpoints (RRQ) y mensajes de admission de llamadas (ARQ) por cualquiera de estos dos medios: por medio de un username/password (RadAuth) o mediante alias/IP (RadAliasAuth). Adicionalmente, si la llamada es desde un endpoint no registrado (y por consiguiente ninguna de las autenticaciones RRQ o ARQ ha sido realizada), La autenticación con mensajes Setup usando RadAliasAuth toma lugar (SetupUnreg).

Ejemplo 3:

```
RadAuth=optional;RRQ,ARQ
RadAliasAuth=required;RRQ,ARQ,SetupUnreg
default=allow
```

7.2. Sección [SimplePasswordAuth]

La sección define el par de userid y password utilizados por el módulo SimplePasswordAuth. Todos los passwords estan encriptados utilizando la utilidad addpasswd.

Modo de uso:

```
addpasswd config section userid password
```

Opciones:

- KeyFilled=123
  Default: 0 Valor por defecto para utilizar un byte de relleno durante la encriptación/desencriptación de passwords.
- CheckID=1
  Default: 0 Verifica si el alias se corresponde con el ID en los tokens.
- PasswordTimeout=120
  Default: -1 El módulo SimplePasswordAuth y todos sus descendientes mantendrán en caché los passwords auténticos. Este campo define el valor del tiempo de caché en segundos 0 significa que no se utilizará caché para passwords, mientras que un valor negativo significa que la caché nunca expira.

7.3. Sección [SQLPasswordAuth]

Autentica endpoints (H.235 enabled) utilizando passwords almacenadas en una base de datos SQL. Esta sección define el driver SQL a utilizar, los parámetros de connexion a la base de datos y las consultas a utilizar para recuperar passwords.

- Driver=MySQL | PostgreSQL
  Default: N/A El driver SQL a utilizar. Actualmente, drivers MySQL and PostgreSQL estan implementados.

- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost Dirección del host SQL server. Puede estar en la forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.
- Database=billing
  Default: billing El nombre de la base de datos a la cual se realiza la conexión.
- Username=gnugk
  El username utilizado para conectarse a la base de datos.
- Password=secret
  El password utilizado para conectarse a la base de datos. Si el password no está especificado, se realiza un intento de conexión a la base de datos sin un password. Si EncryptAllPasswords está habilitada, o una variable KeyFilled esta definida en esta sección, el password esta en forma encriptada y puede ser creado utilizando la utilidad addpasswd.
- CacheTimeout=120
  Default: 0 Este campo define que tiempo el par (alias;password) recuperados desde la base de datos estarán cacheados en la memoria local. Este valor es expresado en segundos. 0 significa no cachear las passwords, mientras que un valor negativo significa que la caché nunca expira (solamente el comando reload refrescará la caché).
- MinPoolSize=5
  Default: 1 Define el número de conexiones activas SQL. Esto permite un mejor rendimiento, puesto que más de una consulta puede ser ejecutada al mismo tiempo. MinPoolSize=1 simula un viejo comportamiento, cuando el acceso a la base de datos es serializada (una consulta a la vez).
- Query=SELECT ...
  Default: N/A Define la consulta SQL utilizada para recuperar passwords H.235 desde la base de datos. La consulta es parametrizada, eso significa que el reemplazo de parámetros es realizado antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de un string (así %%1), especifique %{1} para permitir expansión dentro de expresiones complejas como %{1}123. Para SQLPasswordAuth se han definido dos parámetros:
  - %1 - El alias actual para el cual se consultará el password
  - %2 - El identificador del gatekeeper

  Aquí algunas cadenas de ejemplo:

  ```
  SELECT h235password FROM users WHERE alias = '%1' AND active
  SELECT h235password FROM users WHERE alias = '%1' AND gk = '%2'
  ```

7.4. Sección [RasSrv::RRQAuth]

Especifica la acción sobre los mensajes de recepción (confirm or deny) para el módulo AliasAuth. El primer alias (éste será principalmente un H323ID) del endpoint a ser registrado es buscado en esta sección. Si un parámetro es encontrado, el valor sera aplicado por regla general. Una regla consiste de condiciones separadas por "&". Un registro es aceptado cuando todas las condiciones coinciden.

Sintaxis:

```
<authrules> :=  empty  |  <authrule> "&" <authrules>

  <authrule>  := <authtype> ":" <authparams>
  <authtype>  := "sigaddr" | "sigip"
  <autparams> := [!&]*
```

La notación y significado de <authparams> depende de <authtype>:

- sigaddr - Expresión regular extendida que tiene que corresponderse contra la representación ”PrintOn(ostream)” de la petición. Ejemplo

  ```
  sigaddr:.*ipAddress .* ip = .* c0 a8 e2 a5 .*port = 1720.*
  ```

- sigip - Forma especializada de ’sigaddr’. Escribe la dirección ip utilizando la notación decimal (comunmente utilizada): ”byteA.byteB.byteC.byteD:port”. Ejemplo:

  ```
  sigip:192.168.242.165:1720
  ```

- allow - Siempre aceptar el alias.
- deny - Siempre rechazar el alias.

7.5. Sección [SQLAliasAuth]

Autentica endpoints utilizando reglas almacenadas en una base de datos (Las reglas tienen el mismo formato definido en la sección 7.4 ([RasSrv::RRQAuth])). Esta sección define el driver SQL a ser utilizado, los parámetros de conexión a la base de datoS y las consultas a utilizar para recuperar los patroneS.

- Driver=MySQL | PostgreSQL
  Default: N/A El driver SQL a utilizar. Actualmente, drivers MySQL and PostgreSQL estan implementados.
- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost Dirección del host SQL server. Puede estar en la forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.
- Database=billing
  Default: billing El nombre de la base de datos a la cual se realiza la conexión.
- Username=gnugk
  El username utilizado para conectarse a la base de datos.
- Password=secret
  El password utilizado para conectarse a la base de datos. Si el password no está especificado, se realiza un intento de conexión a la base de datos sin un password. Si EncryptAllPasswords está habilitada, o una variable KeyFilled esta definida en esta sección, el password esta en forma encriptada y puede ser creado utilizando la utilidad addpasswd.

- CacheTimeout=120
  Default: 0 Este campo define que tiempo el par (alias;password) recuperados desde la base de datos estarán cacheados en la memoria local. Este valor es expresado en segundos. 0 significa no cachear las passwords, mientras que un valor negativo significa que la caché nunca expira (solamente el comando reload refrescará la caché).

- MinPoolSize=5
  Default: 1 Define el número de conexiones activas SQL. Esto permite un mejor rendimiento, puesto que más de una consulta puede ser ejecutada al mismo tiempo. MinPoolSize=1 simula un viejo comportamiento, cuando el acceso a la base de datos es serializada (una consulta a la vez).

- Query=SELECT ...
  Default: N/A Define la consulta SQL utilizada para recuperar alias desde la base de datos. La consulta es parametrizada, eso significa que el reemplazo de parámetros es realizado antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de un string (así %%1), especifique %{1} para permitir expansión dentro de expresiones complejas como %{1}123. Para SQLPasswordAuth se han definido dos parámetros:
  - %1 - El alias actual para el cual se consultara el password
  - %2 - El identificador del gatekeeper

  Aquí algunas consultas de ejemplo:

  ```
  SELECT authrule FROM users WHERE alias = '%1' AND active
  SELECT 'sigip:' || host(ip) || port FROM users WHERE alias = '%1'
  ```

7.6. Sección [PrefixAuth]

La sección define la regla de autenticación para el módulo PrefixAuth. Actualmente, solamente ARQs y LRQs pueden ser autorizados por éste módulo.

Primero, se selecciona un prefijo muy específico de acuerdo al campo destinationInfo de la petición recibida. Entonces la petición es aceptada o rechazada de acuerdo a la correspondencia de reglas con la netmask más específica. Si no se encontraron correspondencias de prefijos, y la opción default está especificada, la petición es aceptada o rechazada de acuerdo a eso. De lo contrario esta es rechazada o pasada al siguiente módulo de autenticación de acuerdo a los requerimientos del módulo.

Formato:

```
prefix=authrule[|authrule|...]
```

Syntax:

```
<authrule> :=  <result> <authrule>

  <result>    := deny | allow
  <authrule>  := [!]ipv4:<iprule> | [!]alias:<aliasrule>
```

Donde <iprule> puede ser específicada en notación decimal punteada o en la notación CIDR, <aliasrule> está expresada en espreción regular. Si el signo '!' precede la regla, el sigificado es inverso.

Ejemplo:

```
555=deny ipv4:10.0.0.0/27|allow ipv4:0/0
5555=allow ipv4:192.168.1.1|deny ipv4:192.168.1.0/255.255.255.0
86=deny !ipv4:172.16.0.0/24
09=deny alias:^188884.*
ALL=allow ipv4:ALL
```

En esta configuración, todos los endpoints excepto los de la red 10.0.0.0/27 están permitidos llamar al prefijo 555 (excepto 5555). A Endpoints desde 192.168.1.0/24 no se les permite llamar al prefijo 5555, excepto 192.168.1.1. Endpoints queno pertenecen a 172.16.0.0/24 no se les permite llamar al prefijo 86. Endpoints que tienen un alias que empieza con 188884 no se les permite llamar al prefijo 09. Todas las otras situaciones estan permitidas.

7.7. Sección [RadAuth]

Esta sección define la configuración que posibilitan la autenticación RADIUS basada en H.235 CATs (Cisco Access Tokens) presentes en las peticiones RRQ, ARQ RAS y mensajes Q.931 Setup.

- Servers=SERVER1[:AUTH_PORT[:ACCT_PORT[:SECRET]]];SERVER2[:AUTH_PORT[:ACCT_PORT[:SECRET]]];...
  Default: N/A Los servidores RADIUS a ser utilizados para la autenticación. La lista puede contener un número arbitrario de servidores. El órden de los servidores es importante, porque los servidores serán interrogados por el módulo RADIUS en el órden dado. Si no se indica información del puerto, el número de puerto especificado en DefaultAuthPort será utilizado. Si tampoco se especifica el secret, el shared secret por defecto especificado en SharedSecret es utilizado. Los nombres de los servidores pueden ser direcciones IP o nombres DNS.

  Sample Servers lines:

  ```
  Servers=192.168.1.1
  Servers=192.168.1.1:1645
  Servers=192.168.1.1:1645:1646:secret1
  Servers=radius1.mycompany.com:1812
  Servers=radius1.mycompany.com;radius2.mycompany.com
  Servers=radius1.mycompany.com:1812:1813:secret1;radius2.mycompany.com:1812:1813:secret2
  ```

- LocalInterface=IP_OR_FQDN
  Default: N/A Interfaz de red local particular que el cliente RADIUS debe utilizar para comunicarse con los servidores RADIUS. Este parámetro puede ser útil en máquinas NAT para restringir números de interfaces de red utilizados para la comunicación RADIUS. Por defecto este valor es vacío y permite a la peticione RADIUS ser enviadas a cualquier (la que mejor conviene) interfaz de red. Si usted no esta seguro de lo que está haciendo, es mejor dejar esta opción sin establecer.

- RadiusPortRange=10000-11000
  Default: N/A Por defecto (si esta opción no está establecida) el cliente RADIUS asigna puertos dinámicamente según lo especificado por el sistema operativo. Si usted desea restringir a cliente RADIUS a utilizar solamente puertos de un rango particular, fije este parámetro.

- DefaultAuthPort=PORT_NO
  Default: 1812 Número de puerto por defecto que será utilizado para las peticiones de autenticacion RADIUS (Access-Request packets), if not overriden by Servers attribute.

- SharedSecret=SECRET
  Default: N/A (empty string) Secret utilizada para autenticar éste GnuGk (cliente NAS) con el servidor RADIUS. Este debe ser un password criptográficamente fuerte. Este es el valor por defecto utilizado, si no se colocó un secret específico en el parámetro Servers. Si EncryptAllPasswords está habilitada, o la variable KeyFilled está definida en esta sección, el password está en forma encriptada y debería ser creado utilizando la utilidad addpasswd.

- RequestTimeout=TIMEOUT_MS
  Default: 2000 (miliseconds) Tiempo de descanso (tiempo en milisegundos) para que el servidor RADIUS responda a una petición enviada por el GnuGk. Si no se recibe ninguna respuesta dentro de este período de tiempo, el siguiente servidor RADIUS es consultado.

- IdCacheTimeout=TIMEOUT_MS
  Default: 9000 (miliseconds) Tiempo de descanso (tiempo en milisegundos) para que identificadores de 8-bits de la petición RADIUS sea única. Si el rango de identificadores de 8-bits se agota dentro de este período de tiempo un nuevo socket cliente (socket UDP) es asignado por el módulo RADIUS. Tomemos el ejemplo: Tenemos aproximadamente 60 RRQs/segundo; después de 4 segundos el rango de identificadores de 8-bits se agota; se asigna un nuevo socket; después de los siguientes 4 segundos, el segundo rango de identificadores de 8-bits se agota; un tercer socket es asignado; después del noveno segundo, identificadores del pool 1 estan disponibles otra vez - ... . En general, descanso demasiado largo - demasiados recursos consumidos, descanso demasiado corto - El servidor RADIUS puede tomar los paquetes entrantes como duplicados y por lo tanto borrarlos.

- SocketDeleteTimeout=TIMEOUT_MS
  Default: 60000 (miliseconds) - 60 s Tiempo de descanso (milisegundos) para que sockets RADIUS no utilizados sean cerrados. Este es utilizado conjuntamente con IdCacheTimeout - sockets adicionales creados durante los períodos pesados de carga del GK para atender peticiones entrantes. Estos sockets son cerrados durante los períodos ociosos.

- RequestRetransmissions=NUMBER
  Default: 2 El número de veces en que una simple petición RADIUS es transmitida a cada servidor RADIUS configurado (si no se recibe respuesta). 1 significa no retransmisión, 2 retransmisión simple, ... El método exácto de retransmisión es definido por el atributo RoundRobinServers.

- RoundRobinServers=BOOLEAN
  Default: 1 Método de retransmisión de peticiones RADIUS.

Si se fija en 1, la petición RADIUS es transmitida de la siguiente manera (Hasta que se recibe respuesta):

```
Server #1 Attempt #1, Server #2 Attempt #1, ..., Server #N Attempt #1
...
Server #1 Attempt #RequestRetransmissions, ..., Server #1 Attempt #RequestRetransmissions
```

Si se fija en 0, se preserva la siguiente secuencia:

```
Server #1 Attempt #1, ..., Server #1 Attempt #RequestRetransmissions
...
Server #N Attempt #1, ..., Server #N Attempt #RequestRetransmissions
```

- AppendCiscoAttributes=BOOLEAN
  Default: 0 En el caso de que se fije, un atributo Cisco Vendor Specific RADIUS se incluye en la petición RADIUS (h323-conf-id,h323-call-origin,h323-call-type).

- IncludeTerminalAliases=BOOLEAN
  Default: 1 En el caso de que se fije, un atributo Cisco VSA 'h323-ivr-out' es enviado con una lista de alias con que el endpoint se esta registrando (RRQ.m_terminalAlias). Este atributo se utiliza para proveer un control detallado sobre la lista de alias con los cuales el endpoint es pemitido registrarse. El formato de este atributo es el siguiente:

```
        Cisco-AV-Pair = "h323-ivr-out=terminal-alias:" alias [,alias] [;]
Example:
        Cisco-AV-Pair = "h323-ivr-out=terminal-alias:helpdesk,support,77771;"
```

- UseDialedNumber=BOOLEAN
  Default: 0 Seleccione el tipo de número Called-Station-Id entre el original (según lo marcado por el usuario) - UseDialedNumber=1 - y el reescrito - UseDialedNumber=0.

7.8. Section [RadAliasAuth]

En esta sección se definen parámetros de configuración que permiten la autenticación RADIUS basada en alias de endpoints y/o direcciones IP presentes en las peticiones RAS o en las peticiones Setup Q.931. Este esquema de autenticación es muy utilizado tanto para endpoints registrados en el gatekeeper (ARQ,RRQ) como para llamadas que vienen desde endpoints no registrados(Setup).

- Servers=SERVER1[:AUTH_PORT[:ACCT_PORT[:SECRET]]];SERVER2[:AUTH_PORT[:ACCT_PORT[:SECRET]]];...
  Default: N/A Servidores RADIUS que van a ser utilizados para autenticación de peticiones RAS. Esta lista puede contener un número arbitrario de servidores. El órden de los servidores es importante, porque los servidores serán interrogados por el módulo RADIUS en el órden dado. Si no se indica información del puerto, el número de puerto especificado en DefaultAuthPort será utilizado. Si tampoco se especifica el secret, el shared secret por defecto especificado en SharedSecret es utilizado. Los nombres de los servidores pueden ser direcciones IP o nombres DNS.

  Example:

```
Servers=192.168.3.1:1645;192.168.3.2:1812:1813:mysecret;radius.mycompany.com
```

- LocalInterface=IP_OR_FQDN
  Default: N/A Interfaz de red local particular que el cliente RADIUS debe utilizar

para comunicarse con los servidores RADIUS. Este parámetro puede ser útil en máquinas NAT para restringir números de interfaces de red utilizados para la comunicación RADIUS. Por defecto este valor es vacío y permite a la peticione RADIUS ser enviadas a cualquier (la que mejor conviene) interfaz de red. Si usted no esta seguro de lo que está haciendo, es mejor dejar esta opción sin establecer.

- RadiusPortRange=10000-11000
  Default: N/A Por defecto (si esta opción no está establecida) el cliente RADIUS asigna puertos dinámicamente según lo especificado por el sistema operativo. Si usted desea restringir a cliente RADIUS a utilizar solamente puertos de un rango particular, fije este parámetro.

- DefaultAuthPort=PORT_NO
  Default: 1812 Número de puerto por defecto que será utilizada para las peticiones de autenticacion RADIUS (Access-Request packets), if not overriden by Servers attribute.

- SharedSecret=SECRET
  Default: N/A (empty string) Secret utilizada para autenticar éste GnuGk (cliente NAS) con el servidor RADIUS. Este debe ser un password criptográficamente fuerte. Este es el valor por defecto utilizado, si no se colocó un secret específico en el parámetro Servers. Si EncryptAllPasswords está habilitada, o la variable KeyFilled está definida en esta sección, el password está en forma encriptada y debería ser creado utilizando la utilidad addpasswd.

- RequestTimeout=TIMEOUT_MS
  Default: 2000 (miliseconds) Tiempo de descanso (tiempo en milisegundos) para que el servidor RADIUS responda a una petición enviada por el GnuGk. Si no se recibe ninguna respuesta dentro de este período de tiempo, el siguiente servidor RADIUS es consultado.

- IdCacheTimeout=TIMEOUT_MS
  Default: 9000 (miliseconds) Tiempo de descanso (tiempo en milisegundos) para que identificadores de 8-bits de la petición RADIUS sea única. Si el rango de identificadores de 8-bits se agota dentro de este período de tiempo un nuevo socket cliente (socket UDP) es asignado por el módulo RADIUS. Tomemos el ejemplo: Tenemos aproximadamente 60 RRQs/segundo; después de 4 segundos el rango de identificadores de 8-bits se agota; se asigna un nuevo socket; después de los siguientes 4 segundos, el segundo rango de identificadores de 8-bits se agota; un tercer socket es asignado; después del noveno segundo, identificadores del pool 1 estan disponibles otra vez - ... . En general, descanso demasiado largo - demasiados recursos consumidos, descanso demasiado corto - El servidor RADIUS puede tomar los paquetes entrantes como duplicados y por lo tanto borrarlos.

- SocketDeleteTimeout=TIMEOUT_MS
  Default: 60000 (miliseconds) - 60 s Tiempo de descanso (milisegundos) para que sockets RADIUS no utilizados sean cerrados. Este es utilizado conjuntamente con IdCacheTimeout - sockets adicionales creados durante los períodos pesados de carga del GK para atender peticiones entrantes. Estos sockets son cerrados durante los períodos ociosos.

- RequestRetransmissions=NUMBER
  Default: 2 El número de veces en que una simple petición RADIUS es transmitida a cada servidor RADIUS configurado (si no se recibe respuesta). 1 significa no

retransmisión, 2 retransmisión simple, ... El método exácto de retransmisión es definido por el atributo RoundRobinServers.

- RoundRobinServers=BOOLEAN
  Default: 1 Método de retransmisión de peticiones RADIUS.

  Si se fija en 1, la petición RADIUS es transmitida de la siguiente manera (Hasta que se recibe respuesta):

  ```
  Server #1 Attempt #1, Server #2 Attempt #1, ..., Server #N Attempt #1
  ...
  Server #1 Attempt #RequestRetransmissions, ..., Server #1 Attempt #RequestRetransmissions
  ```

  Si se fija en 0, se preserva la siguiente secuencia:

  ```
  Server #1 Attempt #1, ..., Server #1 Attempt #RequestRetransmissions
  ...
  Server #N Attempt #1, ..., Server #N Attempt #RequestRetransmissions
  ```

- AppendCiscoAttributes=BOOLEAN
  Default: 0 En el caso de que se fije, un atributo Cisco Vendor Specific RADIUS se incluye en la petición RADIUS (h323-conf-id,h323-call-origin,h323-call-type).

- IncludeTerminalAliases=BOOLEAN
  Default: 1 En el caso de que se fije, un atributo Cisco VSA 'h323-ivr-out' es enviado con una lista de alias con que el endpoint se esta registrando (RRQ.m_terminalAlias). Este atributo se utiliza para proveer un control detallado sobre la lista de alias con los cuales el endpoint es pemitido registrarse. El formato de este atributo es el siguiente:

  ```
          Cisco-AV-Pair = "h323-ivr-out=terminal-alias:" alias [,alias] [;]
  Example:
          Cisco-AV-Pair = "h323-ivr-out=terminal-alias:helpdesk,support,77771;"
  ```

- FixedUsername
  Default: N/A Si este parámetro está activado, éste se superpone sobre el valor del atributo User-Name RADIUS para peticiones RADIUS salientes. Eso significa que cada Access-Request se autenticará en cuanto al usuario FixedUsername.

- FixedPassword
  Default: N/A Si no esta activada esta opción, User-Password es una copia de User-Name. Por ejemplo, si User-Name es 'john' entonces el valor de User-Password será también 'john'. Activar este parámetro sustituye este comportamiento y el atributo User-Password será siempre establecido al valor de FixedPassword. Si EncryptAllPasswords está habilitada, o la variable KeyFilled está definida en ésta sección, el password está en forma encriptada y debe ser creada utilizando la utilidad addpasswd.

  Ejemplo 1:

  ```
  (Ni FixedUsername o FixedPassword activadas)
  ```

  Todos los endpoints se autenticarán utilizando sus alias como username y el password. Eso significa, por ejemplo, endpoint 'EP1' se autenticará con el username 'EP1 y el password 'EP1'.

  Ejemplo 2:

```
(FixedUsername no activado)
FixedPassword=ppp
```

Todos los endpoints se autenticarán utilizando como alias y passwords la palabra 'ppp'.

Ejemplo 3:

```
FixedUsername=ppp
FixedPassword=ppp
```

Todos los endpoints se autenticarán utilizando como username la palabra 'ppp' y como password la palabra 'ppp'.

- UseDialedNumber=BOOLEAN
  Default: 0 Seleccione el número Called-Station-Id entre el original (como marcado por el usuario) - UseDialedNumber=1 - y el sustituto - UseDialedNumber=0.

8. Configuración de Accounting

Accounting se denomina a las operaciones que los usuarios realizarán en el sistema del gatekeeper: inicio de una llamada, parada y retroalimentación de la misma; e incluso el inicio y parada del propio gatekeeper.

Las siguientes secciones en el fichero de configuración pueden ser usadas para configurar el accounting.

8.1. Sección [Gatekeeper::Acct]

La sección define una lista de módulos que serán los que realicen el accounting. Las cuentas son para anotar eventos de encendido/apagado del gatekeeper y eventos de inicio/parada/actualización de llamada. Cada módulo de accounting anota eventos recibidos a un módulo de almacenaje específico. Dicho almacenamiento puede ser en texto plano o un servidor RADIUS y muchos más. La configuración es muy similar a la de autenticación de gatekeeper (ver 7.1 ([Gatekeeper::Auth])).

Todos los CDRs serán también enviados al puerto de estado y pueden ser usados por aplicaciones externas.

Sintaxis:

```
acctmod=acciones

<acctmod>  := FileAcct | RadAcct | SQLAcct | ...
<acciones> := <control>[;<evento>,<evento>,...]
<control>  := optional | required | sufficient | alternative
<evento>   := start | stop | connect | update | on | off
```

La lista de eventos le dice al gatekeeper cual de los eventos debe realizar las anotaciones con el módulo de accounting dado (si un tipo de evento está soportado por el módulo):

- start - una llamada ha sido iniciada y un mensaje de Configuración ha sido recibido (solo disponible en modo enrutado),

- connect - una llamada ha sido establecida (solo disponible en modo enrutado),
- update - una llamada está activa y se realizó la actualización periódica para reflejar la duración de la nueva llamada. La Frecuencia de estas actualizaciones está determinada en la sección AcctUpdateInterval variable de 11.1 ([CallTable]),
- stop - una llamada ha sido desconectada o parada (eliminada de la tabla de llamadas de GK),
- on - el gatekeeper ha sido iniciado,
- off - el gatekeeper ha sido apagado.

Un evento anotado por un módulo puede obtener uno de estos tres códigos de resultado: ok, fail, next.

- ok - el evento ha sido anotado correctamente por este módulo,
- fail - el módulo falló al anotar el evento,
- next - el evento no ha sido anotado por este módulo, porque el módulo no está configurado para o no soporta este tipo de evento.

Los módulos de accounting pueden ser apilados para anotar eventos de multiples módulos o para crear configuraciones a prueba de fallos. control bandera para cada módulo, junto con los códigos de resultado, define lo que es el estado final del proceso del vento por el módulo apilado completo. Si el resultado final es failure, algunas acciones especiales se harían en su lugar. Actualmente, si en un evento de llamada iniciada (start) la anotación falla, la llamada es desconectada inmediatamente. Los siguientes estados (banderas) de control están reconocidos:

- required - si el módulo falla al anotar un evento, el estado final es configurado a fallo y el evento es pasado a cualquiera de los restantes módulos,
- optional - el módulo intenta anotar un evento, pero el estado final no es afectado si sale bien o falla (a excepción de cuando el módulo es el último en la lista). El vento es siempre pasado a cualquiera de los módulos restantes,
- sufficient - el módulo determina el estado final. Si un evento es anotado satisfactoriamente, ningún módulo más es procesado. En caso contrario el estado final es configurado a fallo y el evento es pasado a cualquiera del resto de módulos,
- alternative - si el módulo anota un evento satisfactoriamente, ningún módulo más será procesado. En caso contrario el estado final no será modificado y el evento es pasado al resto de módulos.

Los módulos de acconting actualmente soportados:

- FileAcct Un anotador de texto plano CDR. Este emite una línea de estado como líneas CDR a un fichero de texto especificado. Este modulo soporta solo el evento de accounting stop. Las opciones de configuración serán leídas de la sección 8.2 ([FileAcct]).

- RadAcct Este módulo realiza accounting a través de RADIUS. Este soporta todos los tipos de eventos (start, stop, update, on, off). Ver la sección 8.3 ([RadAcct]) para detalles de configuración.
- SQLAcct Este módulo realiza accounting a través de SQL. Este soporta los tipos de evento (start, stop, update). Ver la sección 8.4 ([SQLAcct]) para detalles de configuración.
- default Este es un pseudo-módulo especial - este es usado para configurar el estado final si los módulos precedentes no lo han determinado. El formato es como sigue:

  Sintaxis:

  ```
  default=<estado>[;<evento>,<evento>,...]
  <estado> := accept | fail
  <evento> := start | stop | update | on | off
  ```

Ejemplo de configuración 1 (intenta anotar el inicio/fin de una llamada con un servidor RADIUS, y siempre escribe un CDR a un fichero de texto):

Ejemplo:

```
RadAcct=optional;start,stop
FileAcct=required
```

Ejemplo de configuración 2 (intenta anotar el inicio/fin de una llamada con un servidor RADIUS, si este falla usa un fichero de anotaciones CDR):

Ejemplo:

```
RadAcct=alternative;start,stop
FileAcct=sufficient;stop
default=accept
```

La regla default es requerida aquí para prevenir que la llamada sea rechazada por si la anotación del evento de inicio de RadAcct falla. Si RadAcct retorna el código fail, se pasa al módulo FileAcct. El módulo FileAcct no soporta el evento start, así que retorna el código next. Si no estuviese la regla default, el estado final sería failure, porque ningún módulo ha sido capaz de anotar el evento.

Ejemplo de configuración 3 (siempre anota los eventos de inicio y fin de llamada con un servidor RADIUS, si este falla para el evento de fin de llamada, usa un fichero CDR para almacenar la información de la llamada):

Ejemplo:

```
RadAcct=alternative;start,stop
FileAcct=sufficient;stop
default=fail;start
```

La regla default es opcional aquí. Si RadAcct retorna el código fail para el evento start, el código es pasado al módulo FileAcct. El módulo FileAcct no soporta el evento start, así que retorna el código next. La regla default se asegura de que la llamada

está desconectada si el evento de inicio de llamada puede no haber sido anotado con RadAcct. Sino deseamos almacenar un CDR en un fichero de texto en caso de que el servidor RADIUS esté caído cuando la llamada se desconecte, así podemos ir a por la duración de la llamada dentro del sistema de facturación más tarde.

8.2. Sección [FileAcct]

Este módulo de accounting escribe líneas CDR a un fichero de texto específicado. El formato CDR puede ser uno estándar (el mismo que se puede ver por la interfaz de estado) o uno personalizado (usando la cadena de consulta parametrizada).

- DetailFile=RUTA_COMPLETA_Y_NOMBRE_DE_FICHERO
  Por defecto: N/A Una ruta completa para el fichero de texto plano CDR. Si un fichero con el nombre dado ya existe, los nuevos CDRs serán añadidos al final del fichero.
- StandardCDRFormat=0
  Por defecto: 1 Usa un formato CDR compatible con el formato de la interfaz de estado CDR (1) o crea cadenas CDR personalizadas desde la cadena parametrizada CDRString.
- CDRString=%s|%g|%u|%{Calling-Station-Id}|%{Called-Station-Id}|%d|%c
  Por defecto: N/A Si StandardCDRFormat está desactivado (0) o no se especifica nada, esta cadena parametrizada instruye al gatekeeper de como crear CDRs personalizados. Los parámetros son especificados usando el caracter % y pueden ser una letra (como %n) o más (como %{CallId}). Cualquier caracter más que no sea nombre de parámetro será simplemente copiado al final de la cadena CDR. Los parámetros reconocidos son los siguientes:
  - %g - nombre del gatekeeper
  - %n - número de llamada (no es único después del reinicio del gatekeeper)
  - %d - duración de la llamada (segundos)
  - %c - causa de la desconexión Q.931 (entero decimal)
  - %s - único (para este gatekeeper) identificador de sesión (Acct-Session-Id)
  - %u - H.323 ID de la parte llamada
  - %{CallId} - H.323 identificador de llamada (16 hex digitos de 8-bits)
  - %{ConfId} - H.323 identificador de conferencia (16 hex digitos de 8-bits)
  - %{setup-time} - cadena de hora y fecha para el mensaje de configuración Q.931
  - %{connect-time} - cadena de hora y fecha para un evento de llamada conectada
  - %{disconnect-time} - cadena de hora y fecha para un evento de llamada desconectada
  - %{caller-ip} - dirección IP por la que llama el emisor
  - %{caller-port} - puerto por el que llama el emisor
  - %{callee-ip} - dirección IP por la que recibe la llamada el receptor
  - %{callee-port} - puerto por el que recibe la llamada el receptor
  - %{src-info} - una lista separada por comas de alias del emisor
  - %{dest-info} - una lista separada por comas de alias del receptor
  - %{Calling-Station-Id} - número del emisor
  - %{Called-Station-Id} - número del receptor (reescrito)

- %{Dialed-Number} - número marcado (como es recibido por la parte llamada)

- TimestampFormat=Cisco
  Por defecto: N/A Formato de cadena de fecha y hora impreso en cadenas CDR. Si esta configuración no es especificada, una global de la sección principal del gatekeeper será aplicada.

- Rotate=hourly | daily | weekly | monthly | L... | S...
  Por defecto: N/A Si se configura, el fichero CDR será rotado basado en esta configuración. La rotación horaria activa una rotación por hora, la diaria - una por día, semanal - una por semana y mensual - una por mes. El momento exacto de la rotación es determinado por una combinación de RotateDay y RotateTime. Durante la rotación, un fichero existente será renombrado a CURRENT_FILENAME.YYYYMMDD-HHMMSS, donde YYYYMMDD-HHMMSS es reemplazado con la hora y fecha actuales y un nuevo CDR es anotado a un fichero vacío.
  Además, la rotación por número de CDRs escritos (L...) y por tamaño de fichero (S...) está soportado. El prefijo L especifica un número de líneas CDR escritas, el prefijo S especifica el tamaño de un fichero CDR. Los sufijos k y m pueden ser usados para especificar miles (kilobytes) y millones (megabytes) de bytes. Ver los ejemplos para más detalles.

  Ejemplo 1 - sin rotación:

  ```
  [FileAcct]
  DetailFile=/var/log/gk/cdr.log
  ```

  Ejemplo 2 - rotación cada hora (00:45, 01:45, ..., 23:45):

  ```
  [FileAcct]
  DetailFile=/var/log/gk/cdr.log
  Rotate=hourly
  RotateTime=45
  ```

  Ejemplo 3 - rota cada día a las 23:00 (11PM):

  ```
  [FileAcct]
  DetailFile=/var/log/gk/cdr.log
  Rotate=daily
  RotateTime=23:00
  ```

  Ejemplo 4 - rota cada Domingo (Sunday) a las 00:59:

  ```
  [FileAcct]
  DetailFile=/var/log/gk/cdr.log
  Rotate=weekly
  RotateDay=Sun
  RotateTime=00:59
  ```

  Ejemplo 5 - rota el último día de cada mes:

  ```
  [FileAcct]
  DetailFile=/var/log/gk/cdr.log
  Rotate=monthly
  ```

```
RotateDay=31
RotateTime=23:00
```

Ejemplo 6 - rota cada 10000 CDRs:

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
Rotate=L10000
```

Ejemplo 7 - rota por encima de los 10 kilobytes:

```
[FileAcct]
DetailFile=/var/log/gk/cdr.log
Rotate=S10k
```

8.3. Sección [RadAcct]

Este módulo de accounting envía datos de accounting a un servidor RADIUS. La configuración del módulo es casi lo mismo que para los autenticadores RADIUS (ver 7.7 ([RadAuth]) y 7.8 ([RadAliasAuth]) para más detalles sobre los parámetros).

- Servers=SERVER1[:AUTH_PORT:ACCT_PORT[:SECRET]];SERVER2[:AUTH_PORT:ACCT_PORT[:SECRET]];...
  Por defecto: N/A Servidores RADIUS a los que enviar los datos de accounting. Si no se da puerto de información, el puerto usado se tomará de DefaultAcctPort. Si SECRET no es configurado, por defecto se usará el secreto compartido de SharedSecret. El nombre del servidor puede ser tanto una dirección IP como un nombre DNS.

  Líneas de Servidores de ejemplo:

  ```
  Servers=192.168.1.1
  Servers=192.168.1.1:1645:1646
  Servers=192.168.1.1:1645:1646:secret1
  Servers=radius1.mycompany.com:1812:1813
  Servers=radius1.mycompany.com;radius2.mycompany.com
  Servers=radius1.mycompany.com:1812:1813:secret1;radius2.mycompany.com:1812:1813:secret2
  ```

- LocalInterface=IP_OR_FQDN
  Por defecto: N/A Interfaz de red local en particular que el cliente RADIUS deberá usar para comunicarse con los servidores RADIUS.

- RadiusPortRange=10000-11000
  Por defecto: N/A Por defecto (si esta opción esta activa) el cliente RADIUS asigna los puertos dinámicamente como se especifique por el sistema operativo. Si tu deseas restringir el cliente RADIUS para usar puertos desde un rango particular solo - activa este parámetro.

- DefaultAcctPort=PORT_NO
  Por defecto: 1813 El puerto por defecto será usado por las solicitudes de accounting de RADIUS, si no se especifica por el atributo Servers.

- SharedSecret=SECRET
  Por defecto: N/A (cadena vacía) Un secreto usado para autenticar este GnuGK (cliente NAS) para el servidor RADIUS. Esta debe ser una contraseña criptográficamente sólida. Este es el valor usado por defecto, si no se especifica secret en Servers. Si EncryptAllPasswords está activado, o una variable KeyFilled se definida en esta sección, la contraseña en formato encriptado y debe ser creada usando la utilidad addpasswd.

- RequestTimeout=TIMEOUT_MS
  Por defecto: 2000 (milisegundos) Tiempo de espera (milisegundos) para la respuesta de un servidor RADIUS a una solicitud enviada por GnuGK. Si no se recibe una respuesta dentro de este periodo de tiempo, es consultado el siguiente servidor RADIUS.

- IdCacheTimeout=TIMEOUT_MS
  Por defecto: 9000 (milisegundos) Tiempo de espera (milisegundos) para una solicitud RADIUS con identificadores de 8-bits únicos.

- SocketDeleteTimeout=TIMEOUT_MS
  Por defecto: 60000 (milisegundos) - 60 s Tiempo que se espera para cerrar sockets RADIUS no usados.

- RequestRetransmissions=NUMBER
  Por defecto: 2 Cuantas veces una simple solicitud RADIUS es transmitida a cada servidor RADIUS configurado (si no se recibe una respuesta).

- RoundRobinServers=BOOLEAN
  Por defecto: 1 Método de retransmisión de solicitudes RADIUS.

- AppendCiscoAttributes=BOOLEAN
  Por defecto: 0 Si está activa, los atributos RADIUS específicos de fabricante Cisco son incluídos en las solicitudes RADIUS (h323-conf-id,h323-call-origin, h323-call-type).

- TimestampFormat=ISO8601
  Por defecto: N/A Formato de las cadenas de fecha y hora enviadas en los atributos RADIUS. Si este atributo no está especificado, uno global de la sección principal de gatekeeper será aplicado.

- UseDialedNumber=BOOLEAN
  Por defecto: 0 Selecciona el tipo de número Called-Station-Id entre el original (como fue marcado por el usuario) - UseDialedNumer=1 - y el reescrito - UseDialedNumer=0.

8.4. Sección [SQLAcct]

Este módulo de accounting almacena información de accounting directamente a una base de datos SQL. Muchas opciones de configuración son comunes con otros módulos SQL.

- Driver=MySQL | PostgreSQL
  Por defecto: N/A Manejador de base de datos SQL a usar. Actualmente, los manejadores implementados son MySQL y PostgreSQL.

- Host=DNS[:PORT] | IP[:PORT]
  Por defecto: localhost Dirección de la máquina del servidor SQL. Puede está en forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.

- Database=billing
  Por defecto: billing El nombre de la base de datos a la que conectarse.

- Username=gnugk
  El nombre de usuario usado para conectarse a la base de datos.

- Password=secret
  La contraseña usada para conectarse a la base de datos. Si la contraseña no se especifica, se intentará una conexión a la base de datos sin ninguna contraseña. Si EncryptAllPasswords está activo, o una variable KeyFilled es definida en esta sección, la contraseña está en una forma encriptada y debe ser creada usando la utilidad addpasswd.

- StartQuery=INSERT ...
  Por defecto: N/A Define una consulta SQL usada para insertar un nuevo registro de llamada a la base de datos. La consulta es parametrizada - eso significa que el reemplazo de parámetros se hace antes de que cada consulta sea ejecutada. Los parámetros son precedidos por el caracter % y pueden ser una letra (como %u) o cadenas enteras (como %{src-info}). Especifica %% para añadir un caracter de porcentaje dentro de la cadena de consulta (como %%). Para SQLAcct se pueden utilizar los siguientes parámetros:
  - %g - nombre del gatekeeper
  - %n - número de llamada (no es único después del reinicio del gatekeeper)
  - %d - duración de la llamada (segundos)
  - %c - causa de la desconexión Q.931 (entero decimal)
  - %s - único (para este gatekeeper) identificador de sesión (Acct-Session-Id)
  - %u - H.323 ID de la parte llamada
  - %{gkip} - dirección IP del gatekeeper
  - %{CallId} - H.323 identificador de llamada (16 hex digitos de 8-bits)
  - %{ConfId} - H.323 identificador de conferencia (16 hex digitos de 8-bits)
  - %{setup-time} - cadena de hora y fecha para el mensaje de configuración Q.931
  - %{connect-time} - cadena de hora y fecha para un evento de llamada conectada
  - %{disconnect-time} - cadena de hora y fecha para un evento de llamada desconectada
  - %{caller-ip} - dirección IP por la que llama el emisor
  - %{caller-port} - puerto por el que llama el emisor
  - %{callee-ip} - dirección IP por la que recibe la llamada el receptor
  - %{callee-port} - puerto por el que recibe la llamada el receptor
  - %{src-info} - una lista separada por comas de alias del emisor
  - %{dest-info} - una lista separada por comas de alias del receptor
  - %{Calling-Station-Id} - número del emisor
  - %{Called-Station-Id} - número del receptor (reescrito)

- %{Dialed-Number} - número marcado (como es recibido por la parte llamada)

Consulta de ejemplo:

```
INSERT INTO call (gkname, sessid, username, calling, called)
        VALUES ('%g', '%s', '%u', '%{Calling-Station-Id}', '%{Called-Station-Id}')
```

- StartQueryAlt=INSERT ...
  Por defecto: N/A Define una consulta SQL usada para insertar un nuevo registro de llamada en la base de datos en caso de que StartQuery falle por alguna razón (la llamada ya existe, por ejemplo). La sintaxis y parámetros son los mismos que para StartQuery.

- UpdateQuery=UPDATE ...
  Por defecto: N/A Define una consulta SQL usada para actualizar un registro de llamada en una base de datos con el estado de llamada actual. La sintaxis y parámetros son los mismos que para StartQuery.

  Ejemplo de consulta:

  ```
  UPDATE call SET duration = %d WHERE gkname = '%g' AND sessid = '%s'
  ```

- StopQuery=UPDATE ...
  Por defecto: N/A Define una consulta SQL usada para actualizar el registro de llamada en una base de datos cuando la llamada ha finalizado (desconectada). La sintaxis y parámetros son los mismos que para StartQuery.

  Ejemplo de consulta:

  ```
  UPDATE call SET duration = %d, dtime = '%{disconnect-time}' WHERE gkname = '%g' AND sessid = '%s'
  ```

- StopQueryAlt=INSERT ...
  Por defecto: N/A Define una consulta SQL usada para actualizar un registro de llamada en una base de datos cuando la llamada ha finalizado (desconectada) en caso de que StopQuery falle (porque el registro de llamada no exista ya, por ejemplo). La sintaxis y parámetros son los mismos que para StartQuery.

  Ejemplo de consulta:

  ```
  INSERT INTO call (gkname, sessid, username, calling, called, duration)
          VALUES ('%g', '%s', '%u', '%{Calling-Station-Id}', '%{Called-Station-Id}', %d)
  ```

- TimestampFormat=MySQL
  Default: N/A Formato de cadena de fecha y hora usada en consultas. Si este atributo no es especificado, uno global de la sección principal de gatekeeper será aplicado.

9. Configuración de Vecinos

9.1. Sección [RasSrv::Neighbors]

Si el destino ce un ARQ es desconocido, el gatekeeper envía LRQs hacia sus vecinos para preguntarles si ellos tienen el endpoint destino. Un vecino es seleccionado si uno de sus prefijos se corresponde con el destino o si este tiene el prefijo "*". Se pueden especificar más de un prefijo. Usted puede utilizar caracteres especiales como "." y "!" para hacer corresponder ciertos comodines (wildcard) y deshabilitar un prefijo específico.

Recíprocamente, el gatekeeper contestará solamente los LRQs enviados desde los vecinos definidos en esta sección. Si usted especifica un prefijo vacío, ningún LRQ será enviado hacia ese vecino, pero el gatekeeper aceptará LRQs que vengan desde éste. Para especificar un prefijo vacío se añade un punto y coma a la entrada del vecino. Ejemplo:

```
GK1=192.168.0.5;
```

Si usted no pone el punto y coma, LRQs serán siempre enviados hacia este vecino.

El campo password es utilizado pata autenticar LRQs desde ese vecino. Ver la sección 7.1 ([Gatekeeper::Auth]) para más información.

El manejo de los vecinos ha cambiado significativamente desde la versión 2.0 hasta la version 2.2. Actualmente los vecinos puedes ser configurados de dos maneras - de la forma anteriormente conocida y de una nueva manera.

Configuración en el formato anterior:

```
GKID=ip[:port;prefixes;password;dynamic]
```

Ejemplo:

```
GK1=192.168.0.5;*
GK2=10.0.1.1:1719;035,036;gk2
GK3=gk.citron.com.tw;;gk3;1
```

Configuración en el nuevo formato:

```
GKID="GnuGK"| ÇiscoGK"| ÇlarentGK"| "GlonetGK"
```

Ejemplo:

```
[RasSrv::Neighbors]
GK1=CiscoGK
GK2=GnuGK

[Neighbor::GK1]
GatekeeperIdentifier=GK1
Host=192.168.1.1
SendPrefixes=02
AcceptPrefixes=*
ForwardLRQ=always

[Neighbor::GK2]
GatekeeperIdentifier=GK2
Host=192.168.1.2
SendPrefixes=03,0048
AcceptPrefixes=0049,001
ForwardHopCount=2
ForwardLRQ=depends
```

El nuevo formato especifica dentro de la sección [RasSrv::Neighbors] solamente tipos de gatekeeper y las configuraciones para cada vecino se ubican en una sección separada.

9.2. Sección [RasSrv::LRQFeatures]

Define algunas características de LRQ y LCF.

- NeighborTimeout=1
  Default: 2 Tiempo en segundos para esperar respuestas desde los vecinos. Si no hay respuesta desde todos los vecinos después de este tiempo, el gatekeeper contestará con un ARJ al endpoint que envió el ARQ.

- ForwardHopCount=2
  Default: N/A Si el gatekeeper recive un LRQ en el que el destino es desconocido, éste reenviará este mensaje hacia sus vecinos. Cuanod el gatekeeper recibe un LRQ y decide que el mensaje debe ser reenviado hacia otro gatekeeeper, éste primero decrementa el campo hopCount de el LRQ. Si hopCount ha llegado a 0, el gatekeeper no reenviará el mensaje. Esta opción define el número de gatekeepers a través de los cuales un mensaje LRQ puede propagarse. Tenga en cuenta que esto solo afecta al remitente del LRQ, no al receptor. Este ajuste puede ser superpuesto con la configuración de un vecino particular.

- AlwaysForwardLRQ=1
  Default: 0 Forzar a que el gatekeeper reenvie un LRQ incluso si no hay hopCount en el LRQ. Para evitar LRQ repetidos, usted debe utilizar esta opción muy cuidadosamente. Esta opción es utilizada solamente para para un estilos anteriores de configuración de vecinos (Version 2.0), el nuevo estilo lee las configuraciones desde la sección de configuración del vecino específico.

- AcceptForwardedLRQ=1
  Default: 1 Definir si aceptar o no un LRQ reenviado desde los vecinos. Esta configuración puede ser superpuesta o deshabilitada con la configuración de un vecino particular.

- IncludeDestinationInfoInLCF=0
  Default: 1 El gatekeeper responde LCFs conteniendo los campos destinationInfo y destinationType, los alias y el tipo de terminal del endpoint destino. El gatekeeper vecino puede entonces guardar la información para suprimir LRQs atrasados. Sin embargo, algunos proveedores de gatekeepers emplean mal esta información, de esta manera resultan en problemas de interoperabilidad. Solamente desactive esta opción si usted encuentra problemas en la comunicación con un gatekeeper de terceros.

- ForwardResponse=0
  Default: 0 Si el gatekeeper reenvió los mensajes LRQ recibidos, éste puede decidir entre recibir la repuesta LCF o permitir que ésta regrese directamente hacia el gatekeeper que originó en LRQ. Establezca esta opción en 1, si el gatekeeper necesita recibir respuestas LCF de los LRQs reenviados. Este ajuste puede ser superpuesto o deshabilitado con la configuración de un vecino particular.

- ForwardLRQ=always | never | depends
  Default: depends Esta configuración determina si el LRQ recibido debería o no ser reenviado . always reenvía LRQs de manera incondicional, never bloquea los LRQ reenviados, depends le indica al gatekeeper que reenviará LRQ solamente si su contador de saltos (hop count) es mayor a 1. Este ajuste puede ser superpuesto o deshabilitado con la configuración de un vecino particular.

9.2.1. Sección [Neighbor::...]

Las secciones que empiezan con [Neighbor:: son para las configuraciones especificas de un vecino.

- GatekeeperIdentifier=GKID
  Default: N/A Identificador del Gatekeeper para este vecino. Si esta opción no se especifica, el identificador es tomado de la segunda parte del nombre de esta sección Neighbor::.

- Host=192.168.1.1
  Default: N/A Una dirección IP para este vecino.

- Password=secret
  Default: N/A Un password que se utilizará pata validar los crypto tokens recibidos desde los LRQs entrantes. Esto aún no está implementado.

- Dynamic=0
  Default: 0 1 significa que la dirección IP para este vecino puede cambiar.

- SendPrefixes=004,002:=1,001:=2
  Default: N/A Una lista de prefijos que este vecino espera recibir para los LRQs. Si se especifica '*', Los LRQs siempre seran enviados hacia este vecino. Se puede asignar una prioridad a cada prefijo para cada vecino (usando la sintaxis :=), de esta manera en caso de que hayan demasiados LCF recibidos de demasiados vecinos, aquel con la prioridad más alta será elegido para rutear la llamada. Uno también puede dirigir el gatekeeper para que envíe LRQs hacia este vecino basándose en el tipo de alias:
  SendPrefixes=h323_ID,dialedDigits,001

- AcceptPrefixes=*
  Default: * Una lista de prefijos que el gatekeeper debería aceptar en los LRQs recibidos desde este vecino. Si se especifica '*' , todos los LRQs serán aceptados desde este vecino. Uno puede también dirigir al gatekeeper para que acepte LRQ desde este vecino basándose en el tipo de alias:
  AcceptPrefixes=dialedDigits

- ForwardHopCount=2
  Default: N/A Si el gatekeeper recibe un LRQ en el que el destino es desconocido, éste podría reenviar este mensaje hacia sus vecinos. Cuando el gatekeeper recibe un LRQ y decide que el mensaje debe ser reenviado hacia otro gatekeeper, éste primero decrementa el campo hopCount del LRQ. Si hopCount ha llegado a 0, el gatekeeper no reenviará el mensaje. Esta opción define el número de gatekeepers a través de los cuales un LRQ puede propagarse. Tenga en cuenta que esto solo afecta al remitente del LRQ, no al receptor.

- AcceptForwardedLRQ=1
  Default: 1 Definir si aceptar o no un LRQ reenviado desde este vecino.

- ForwardResponse=0
  Default: 0 Si el gatekeeper reenvió los mensajes LRQ recibidos, éste puede decidir entre recibir la repuesta LCF o permitir que ésta regrese directamente hacia el

gatekeeper que originó en LRQ. Establezca esta opción en 1, si el gatekeeper necesita recibir respuestas LCF de los LRQs reenviados.

- ForwardLRQ=always | never | depends
  Default: depends Esta configuración determina si el LRQ recibido debería o no ser reenviado . always reenvía LRQs de manera incondicional, never bloquea los LRQ reenviados, depends le indica al gatekeeper que reenviará LRQ solamente si su contador de saltos (hop count) es mayor a 1. Este ajuste puede ser superpuesto o deshabilitado con la configuración de un vecino particular.

10. Configuración Por-Terminal

Para las opciones del fichero de configuración estándar, atributos de configuración por terminal pueden ser especificados en el fichero de configuración. La sintaxis es como sigue:

10.1. Sección [EP::...]

```
[EP::ALIAS]
Key Name=Value String
```

ALIAS es reemplazado con un alias actual para un terminal para que la configuración sea aplicada. Actualmente, están reconocidas las siguientes opciones:

- Capacity=10
  Por defecto: -1 Capacidad de llamadas para un terminal. No más de Capacity llamadas concurrentes serán enviadas a este terminal. En caso de puertas de enlace (gateways), si más de una puerta de enlace corresponde a un número marcado, la llamada será enviada a la primera puerta de enlace disponible (que tenga suficiente capacidad).

- GatewayPriority=1
  Por defecto: 1 Aplicado solo a puertas de enlace (gateways). Permite la prioridad basada en casos de encaminado, cuando más de una puerta de enlace corresponde a un número marcado. El valor más pequeño es la prioridad más alta asignada a una puerta de enlace. Una llamada es encaminada a la primera puerta de enlace disponible (que tenga capacidad disponible) con la prioridad más alta (los valores más bajos en GatewayPriority).

- GatewayPrefixes=0048,0049,0044
  Por defecto: N/A Prefijos adicionales para esta puerta de enlace. Aplicado solo a puertas de enlace. Los caracteres especiales . y ! pueden ser usados aquí para la correspondencia con cualquier dígito y desactivar el prefijo (respectivamente).

Ejemplo:

```
[RasSrv::PermanentEndpoints]
192.168.1.1=gw1;48
192.168.1.2=gw2;48,!4850,!4860,!4869,!4888

[EP::gw1]
```

```
Capacity=60
GatewayPriority=1

[EP::gw2]
Capacity=30
GatewayPriority=2
```

En este ejemplo, la llamada será enviada a la puerta de enlace gw1 a menos que su capacidad esté completamente utilizada (60 llamadas concurrentes) y entonces pase a la puerta de enlace gw2.

11. Configuración Avanzada

11.1. Sección [CallTable]

- GenerateNBCDR=0
  Default: 1 Genere CDRs para llamadas que vienen desde zonas vecinas. La dirección IP y el ID del endpoint que esta llamando se presenta como una cadena vacía. Esto es frecuentemente utilizado para propósitos de debug.
- GenerateUCCDR=0
  Default: 0 Genere CDRs para llamadas que son desconectadas (unconnected). Es es frecuentemente utilizado para propósitos de debug. Tenga presente que una llamada se considera desconectada (unconnected) solamente si el gatekeeper utiliza modo de ruteo (routed mode) y no recibe un mensaje de conexión Q.931. En modo directo (direct mode), siempre una llamada es considerada conectada (connected).
- DefaultCallDurationLimit=3600
  Default: 0 Límite máximo de tiempo (por defecto) que durará una llamada (en segundos). Establesca esta variable a 0 para deshabilitar esta característica y no limitar la duración de las llamadas.
- AcctUpdateInterval=60
  Default: 0 Un intervalo de tiempo (en segundos) para actualización de accounting (accounting update), dejando un registro para cada llamada en progreso. Detalles exactos de actualización de accounting (accounting update) dependen del módulo de registro de accounting elegido (ver la sección 8.1 ([Gatekeeper::Acct])). En general, la actualización del accounting (accounting update) es con la finalidad de tener un servicio de respaldo con el incremento de la duración de la llamada, para llamadas establecidas o conectadas (connected calls). El valor por defecto 0 le indica al gatkeeper que no realize actualización de accounting en absoluto. Por favor tenga presente que establecer períodos cortos de tiempo podría disminuír el rendimiento del GK.
- TimestampFormat=Cisco
  Default: RFC822 Formato de presentación de las cadenas impresas dentro del los CDRs.

11.2. Sección [Endpoint]

El gatekeeper puede trabajar como un endpoint registrandose con otro gatekeeper. Con esta característica, usted puede construír facilmente jerarquías de gatekeepers. Esta sección define las características de un endpoint para el gatekeeper.

- Gatekeeper=10.0.1.1
  Default: no Defina un gatekeeper padre (parent gatekeeper) para el endpoint(el gatekeeper), con el cual se va a registrar. No trate de registrarse con usted mismo, a menos de que usted desee confundirse. Para dehabilitar esta característica, configure este campo para que sea no.

- Type=Gateway
  Default: Gateway Defina el tipo de terminal para el endpoint. Los valores válidos son Gateway o Terminal.

- H323ID=CitronProxy
  Default: <Name> Especifique el o los alias H.323 ID para el endpoint. Múltiples alias pueden separarse con comas.

- E164=18888600000,18888700000
  Default: N/A Define el alias E.164 (dialedDigits) para el endpoint. Múltiples alias pueden separarse con comas.

- Password=123456
  Default: N/A Especifique un password para que sea enviado al gatekeeper padre. Todas las peticiones RAS contendrán el password en el campo cryptoTokens (MD5 & HMAC-SHA1-96) y en el campo tokens (CAT). Para enviar peticiones RAS sin los camposcryptoTokens y tokens, establesca el password como vacío. Si la opción EncryptAllPasswords está habilitada, o la variable KeyFilled esta definida en esta sección, el password está en forma encriptada y deberá ser creada utilizando la utilidad addpasswd.

  Además, el password es también utilizado en mensajes LRQs enviados a gatekeepers vecinos.

- Prefix=188886,188887
  Default: N/A Registre los prefijos utilizados con el gatekeeper padre. Solamente tiene efecto cuando el tipo de terminal es Gateway.

- TimeToLive=900
  Default: 60 Sugiera un valor time-to-live (en segundos)para el proceso de registro. Tenga presente que el real contador para time-to-live es asignado por el gatekeeper padre en el mensaje RCF contestando a la petición RRQ.

- RRQRetryInterval=10
  Default: 3 Defina un intervalo de reintento en segundos para reenviar un RRQ si no se ha recibido respuesta desde el gatekeeper padre. Este intervalo es duplicado con cada falla, a un máximo de interrupción de RRQRetryInterval * 128.

- ARQTimeout=2
  Default: 2 Define un valor de espera en segundos para los ARQs.

- UnregisterOnReload=1
  Default: 0 Define si el gatekeeper hijo desregistra y re-registra con su gatekeeper padre cuando recibe el comando Reload.

- NATRetryInterval=60
  Default: 60 Tiempo que se debe esperar antes de tratar de reconectar el TCP NAT signalling socket (en segundos). Esto puede pasar por cualquiera de estas razones:la conexión no puede ser establecida o ésta ha sido rota.

- NATKeepaliveInterval=86400
  Default: 86400 Define que tan a menudo es refrescada la conexión de señalización TCP NAT con un gatekeeper padre. Puesto que las cajas NAT usualmente mantienen TCP mappings solamente por un tiempo definido, es bueno establecer éste con un valor un poco mas pequeñoque el de la caja NAT. El refresco de la conexión es realizado enviando un mensaje Q.931 IncomingCallProceeding. Si su NAT realiza traducción de puertos TCP, usted podría necesitar establecer éste a un valor cercano a 60 segundos.

- Discovery=0
  Default: 1 Decida se descubre o no el gatekeeper padre enviando primero un GRQ.

- UseAlternateGK=0
  Default: 1 Habilite la característica de alternar gatekeepers. Si los mensajes GRJ/GCF/RFC recibidos desde el gatekeeper padre contienen una lista de gatekeepers alternos, esta información es almacenada y puede ser utilizada después para registrarse con otro gatekeeper en caso de cualquier falla. Si usted no desea usar esta característica, establezca esta variable a 0.

- GatekeeperIdentifier=ParentGK
  Default: Not set Defina este parámetro si usted desea aceptar solamente aquellos gatekeepers que correspondan a este identificador de gatekeeper. Es muy utilizado con GRQ discovery y puede prevenir una asignación de gatekeeper accidental. No establezca esta variable, si a usted no le importa acerca de identificadores de gatekeeper o si usted utiliza gatekeepers alternos que pueden tener diferentes identificadores de gatekeeper establecidos.

- EndpointIdentifier=OpenH323GK
  Default: Not set Establezca esta característica si usted desea utilizar un identificador de endpoint específico para este gatekeeper hijo. Si esta opción no esta establecida (por defecto), el gatekeeper padre asignará el identificador en un mensaje GCF/RCF.

11.3. Sección [CTI::Agents]

Esta sección permite la configuración de las llamadas colas virtuales (virtual queues) para permitir distribuciónde llamadas entrantes por una aplicación externa, mediante el puerto de estado. Una cola virtual tiene un alias H.323 que puede ser llamado como un endpoint.

En el arrivo de un mensaje ARQ a una cola virtual, el gatekeeper señala una RouteRequest en el puerto de estado y espera a que una aplicación externa responda ya sea con un RouteReject (entonces el ARQ será rechazado) o con un RouteToAlias/RouteToGateway el cual conduce al ARQ a ser reescrito de este modo la llamada será ruteada hacia el alias (eg. call center agent) especificado por la aplicación externa.

Si no se recibe ninguna respuesta después de un período de tiempo, la llamada es finalizada.

Usted puede definir colas virtuales de tres maneras:

- nombre de alias exacto - Una lista de alias es asignada. Si un alias destino ARQ corresponde a uno de estos nombres, la cola virtual es activada,

- prefijo - Una lista de prefijos es dada. Si un alias destino ARQ inicia con uno de estos prefijo, la cola virtual es activada,
- expresión regular - Una expresión regular es dada. Si un alias destino ARQ corresponde a la expresión, la cola virtual es activada.

Ver la sección de monitoreso (monitoring section) para detalles sobre los mensajes y respuestas.

- VirtualQueueAliases
  Default: none Este parámetro define una lista de alias H.323 para las colas virtuales (Utilizado con vqueue RoutingPolicy).

  Ejemplo:

  ```
  VirtualQueueAliases=sales,support
  ```

- VirtualQueuePrefixes
  Default: none Este parámetro define una lista de prefijos para las colas virtuales (Utilizado con vqueue RoutingPolicy).

  Ejemplo:

  ```
  VirtualQueuePrefixes=001215,1215
  ```

- VirtualQueueRegex
  Default: none Este parámetro deine una expresión regular para las colas virtuales (Utilizado con vqueue RoutingPolicy).

  Ejemplo (números que empiezan con 001215 o 1215):

  ```
  VirtualQueueRegex=^(001|1)215[0-9]*$
  ```

- RequestTimeout
  Default: 10 Tiempo de espera en segundos para que la aplicación externa responda el RouteRequest. Si no se recibe ninguna respuesta durante este tiempo un ARJ será enviado hacia el enpoint que llama (caller).

11.4. Sección [SQLConfig]

Carge configuraciones del gatekeeper desde una base de datos SQL (adicionalmente a las configuraciones leidas desde el archivo de configuración). Una ConfigQuery genérica puede ser utilizada para leer casi todas las configuraciones desde la base de datos y/o una de [RasSrv::RewriteE164], [RasSrv::PermanentEndpoints], [RasSrv::Neighbors], [RasSrv::GWPrefixes] consultas pueden ser utilizadas para cargar configuraciones particulares. Los valores leídos desde la base de datos SQL tiene preferencia sobre las configuraciones encontradas en el archivo de configuración.

- Driver=MySQL | PostgreSQL
  Default: N/A El driver SQL a utilizar. Actualmente, drivers MySQL and PostgreSQL estan implementados.
- Host=DNS[:PORT] | IP[:PORT]
  Default: localhost Dirección del host SQL server. Puede estar en la forma de DNS[:PORT] o IP[:PORT]. Como sql.mycompany.com o sql.mycompany.com:3306 o 192.168.3.100.

- Database=billing
  Default: billing El nombre de la base de datos a la cual se realiza la conexión.

- Username=gnugk
  El username utilizado para conectarse a la base de datos.

- Password=secret
  El password utilizado para conectarse a la base de datos. Si el password no está especificado, un intento de conexión a la base de datos sin un password es realizada. Si EncryptAllPasswords está habilitada, o una variable KeyFilled esta definida en esta sección, el password esta en forma encriptada y puede ser creado utilizando la utilidad addpasswd.

- ConfigQuery=SELECT ...
  Default: N/A Defina una consulta SQL utilizada para leer configuraciones del gatekeeper desde la base de datos. La consulta es parametrizada, lo que significa que el remplazo de parámetros ocurre antes de que la consulta es ejecutada. Los parámetros están denotados por las cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para ConfigQuery solamente está definido un parámetro:

  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de tres columnas:

  - columna con índice 0 - nombre de la sección de configuración (config section name)
  - columna con índice 1 - nombre de la opción (config key)
  - columna con índice 2 - valor de la opción (config value)

  Ejemplos de consultas:

  ```
  ConfigQuery=SELECT secname, seckey, secval FROM sqlconfig WHERE gk = '%1'
  ConfigQuery=SELECT '[RasSrv::RRQAuth]', alias, rule FROM rrqauth WHERE gk = '%1'
  ```

- RewriteE164Query=SELECT ...
  Default: N/A Defina una consulta SQL para recuperar desde la base de datos reglas de reescritura (rewrite rules) para la sección [RasSrv::RewriteE164]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para RewriteE164Query solamente está definido un parámetro:

  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de dos columnas:

  - columna con índice 0 - nombre de la regla de reescritura (rewrite rule key)
  - columna con índice 1 - valor de la regla de reescritura (rewrite rule value)

  Ejemplos de consulta:

  ```
  RewriteE164Query=SELECT rkey, rvalue FROM rewriterule WHERE gk = '%1'
  ```

- NeighborsQuery=SELECT ...
  Default: N/A Defina una consulta SQL para recuperar desde la base de datos entradas vecinas (neighbor entries) para la sección [RasSrv::Neighbors]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para NeighborsQuery está definido un parámetro:
  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de seis columnas:
  - columna con índice 0 - nombre del vecino (identificador)
  - columna con índice 1 - dirección IP del vecino
  - columna con índice 2 - número de puerto del vecino
  - columna con índice 3 - prefijos opcionales (separados por comas)
  - columna con índice 4 - password opcional
  - columna con índice 5 - IP dinámica opcional (optional dynamic IP flag)

  Ejemplos de consultas:

  ```
  NeighborsQuery=SELECT nid, nip, nport, npfx, NULL, 0 FROM neighbor WHERE gk = '%1'
  ```

- PermanentEndpointsQuery=SELECT ...
  Default: N/A Defina una consulta SQL para recuperar endpoints permanentes desde la base de datos para la sección [RasSrv::PermanentEndpoints]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para PermanentEndpointsQuery solamente está definido un parámetro:
  - %1 - El identificador del gatekeeper

  Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de cuatro columnas:
  - columna con índice 0 - dirección IP del endpoint permanente
  - columna con índice 1 - número de puerto del endpoint permanente
  - columna con índice 2 - alias del endpoint permanente
  - columna con índice 3 - prefijos opcionales del endpoint permanente (separados por comas)

  Ejemplos de consultas:

  ```
  PermanentEndpointsQuery=SELECT peip, 1720, pealias, NULL FROM permanentep WHERE gk = '%1'
  ```

- GWPrefixesQuery=SELECT ...
  Default: N/A Defina una consulta SQL para recuperar prefijos de gateways desde la base de datos para la sección [RasSrv::GWPrefixes]. La consulta es parametrizada, lo que significa que el reemplazo de parámetros ocurre antes de que cada consulta es

ejecutada. Los parámetros están denotados por cadenas %1, %2, ... Especifique %% para insertar un caracter de porcentaje antes de un dígito dentro de una cadena (así %%1), especifique %{1} para pemitir expansión dentro de expesiones complejas como %{1}123. Para GWPrefixesQuery solamente está definido un parámetro:

- %1 - El identificador del gatekeeper

Se espera que la consulta devuelva cero o mas filas de datos, en donde cada fila consiste de dos columnas:

- columna con índice 0 - alias del gateway
- columna con índice 1 - prefijos del gateway (separados por comas)

Ejemplos de consultas:

```
GWPrefixesQuery=SELECT gwalias, gwpfx FROM gwprefix WHERE gk = '%1'
```

12. Monitoreando el Gatekeeper

12.1. Puerto de Estado

El puerto de estado es una interfaz externa para monitorear y controlar el gatekeeper. El gatekeeper arrojará mensajes sobre las llamadas actuales hacia todos los clientes conectados y puede recibir comandos mediante esta interfaz.

Los mensajes enviados por el gatekeeper hacia el puerto de estado estan agrupados dentro de tres Niveles de rastreo de salida:

- Nivel 0

  Notificacione se recarga (Reload) y respuesta directa a comandos ejecutados.

- Nivel 1

  Notificacione se recarga (Reload), respuesta directa a comandos ejecutados, CDRs y Route Requests.

- Nivel 2

  Todas las salidas (Notificacione se recarga (Reload), respuesta directa a comandos ejecutados, CDRs, Route Requests,RAS, ...). Este es el nivel de salida por defecto.

El cliente que este conectado hacia el puerto de estado puede seleccionar el nivel de salida (output level) que él desee.

La interfaz es un simple puerto TCP (por defecto: 7000), usted puede conectarse hacia éste mediante telnet u otro cliente. Un ejemplo de un cliente diferente es el Java GUI, aka GkGUI. Otro ejemplo es la aplicación Automatic Call Distribution, aka GnuGk ACD.

12.1.1. Areas de aplicación

Depende de usted lo que haga con el poder del puerto de estado, pero aqui hay algunas ideas:

- Monitoreo de llamadas
- Monitoreo de endpoints registrados
- Interfaz Gráfica de usuario (GUI)

  Ver GkGUI.
- Ruteo de llamadas

  Ver GnuGk ACD.
- Aplicaciones de facturación

  Analice de los mensajes CDR messages y administrelos luego desde una aplicación de facturación.
- Interfaz con aplicaciones externas

  Si usted no desea publicar el código fuente de alguna característica adicional, simplemente publique la funcionalidad central y únalo mediante el puerto de estado y mantenga la parte externa como privada.

12.1.2. Ejemplos:

Supongamos que usted esta interesado en los CDRs (call details records) y quiere un proceso que trabaje en modo batch en intervalos regulares.

Aqui está un script simple escrito en Perl (gnugk_cdr.pl) que inicia el gatekeeper y además conecta un simple cliente para la Status Interface y registra los CDRs dentro de un logfile. Usted debería modificar un poco este script para ajustarlo a sus necesidades.

```
#!/usr/bin/perl
# sample program that demonstrates how to write the CDRs to a log file
use strict;
use IO::Socket;
use IO::Handle;

my $logfile = "/home/jan/cdr.log";      # CHANGE THIS
my $gk_host = "localhost";
my $gk_port = 7000;
my $gk_pid;

if ($gk_pid = fork()) {
        # parent will listen to gatekeeper status
        sleep(1);       # wait for gk to start
        my $sock = IO::Socket::INET->new(PeerAddr => $gk_host, PeerPort => $gk_port, Proto => 'tcp')
        if (!defined $sock) {
                die "Can't connect to gatekeeper at $gk_host:$gk_port";
        }
```

```
        $SIG{HUP} = sub { kill 1, $gk_pid; };   # pass HUP to gatekeeper
        $SIG{INT} = sub { close (CDRFILE); kill 2, $gk_pid; };  # close file when terminated

        open (CDRFILE, ">>$logfile");
        CDRFILE->autoflush(1);  # don't buffer output
        while (!$sock->eof()) {
                my $msg = $sock->getline();
                $msg = (split(/;/, $msg))[0];   # remove junk at end of line
                my $msgtype = (split(/\|/, $msg))[0];
                if ($msgtype eq "CDR") {
                        print CDRFILE "$msg\n";
                }
        }
        close (CDRFILE);
} else {
        # child starts gatekeeper
        exec("gnugk");
}
```

Tenga siempre presente que éste es justamente un ejemplo para mostrar el uso del puerto de estado. Usted puede utilizar el módulo FileAcct para registrar los CDRs en un sistema en producción.

12.1.3. GUI para el gatekeeper

Hay algunas Graphical User Interface (GUI) frontends para el Gatekeeper.

- Java GUIDesarrollado por Jan Willamowius. Usted puede monitorear los registros y llamadas que están en el gatekeeper. Un clic derecho sobre un botón le muestra a usted un menú desplegable con información de dicho endpoint.

  Esta GUI trabaja con Java 1.0 soportado por muchos navegadores web. Por razones de seguridad en GUI debe ser ejecutado como una aplicación standalone o administrada por un servidor web sobre el mismo número de IP del gatekeeper (usted no puede ejecutar esta GUI como un applet mediante el archivo local).

  La aplicación está disponible en <http://www.gnugk.org/h323gui.html>

- GkGUIUna nueva aplicación standalone Java desarrollado por *Citron Network Inc.* <http://www.citron.com.tw/> Este requiere Java 1.4. Entre las nuevas características se incluyen:

  - Monitorea múltiples gatekeepers simultáneamente.
  - Dos modos de vista: Button List y Tree List.
  - Call Detail Record(CDR) y estadísticas.
  - GK Status Log.
  - Colores diferentes para diferentes tipos de endpoints.
  - Modifica la configuración del gatekeeper.
  - Forza a desregistrar endpoints.
  - Guarda e imprime el estado del log y del CDR.

  El GkGUI esta lanzada bajo GNU General Public License, disponible en <http://www.gnugk.org/h323develop.html#java>

12.2. Comandos (Referencia)

Esta sección muestra todos los comandos que usted puede enviar al puerto de estado (manualmente o mediante una aplicación externa). Todos los comandos son case-insensitive. Pero algunos parámetros pueden ser case-sensitive.

El comando help o h le mostrará una lista de todos los comando disponibles.

- Reload Recarga o reinicia la configuración.
- Version, v Muestra la versión e información del OS del gatekeeper.
- Statistics, s Muestra información estadística del gatekeeper.

  Ejemplo:

```
Statistics
-- Endpoint Statistics --
Total Endpoints: 21  Terminals: 17  Gateways: 4  NATed: 2
Cached Endpoints: 1  Terminals: 1  Gateways: 0
-- Call Statistics --
Current Calls: 1 Active: 1 From Neighbor: 0 From Parent: 0
Total Calls: 1539  Successful: 1076  From Neighbor: 60  From Parent: 5
Startup: Fri, 21 Jun 2002 10:50:22 +0800   Running: 11 days 04:22:59
;
```

- PrintAllRegistrations, r, ? Muestra todos los endpoints registrados.

  Formato:

```
AllRegistrations
RCF|IP:Port|Aliases|Terminal_Type|EndpointID
...
Number of Endpoints: n
;
```

  Ejemplo:

```
AllRegistrations
RCF|10.1.1.10:1720|800:dialedDigits=Wei:h323_ID|terminal|1289_endp
RCF|10.0.1.43:1720|613:dialedDigits=Jacky Tsai:h323_ID|terminal|1328_endp
RCF|10.0.1.55:1720|705:dialedDigits=Sherry Liu:h323_ID|terminal|1333_endp
Number of Endpoints: 3
;
```

- PrintAllRegistrationsVerbose, rv, ?? Muestra el detalle de todos los endpoints registrados.

  Formato:

```
AllRegistrations
RCF|IP:Port|Aliases|Terminal_Type|EndpointID
Registration_Time C(Active_Call/Connected_Call/Total_Call) <r>
[Prefixes: ##] (gateway only)
...
Number of Endpoints: n
;
```

  Ejemplo:

```
AllRegistrations
RCF|10.0.1.8:1720|Accel-GW2:h323_ID|gateway|1322_endp
Wed, 26 Jun 2002 16:40:03 +0800 C(1/5/33) <1>
Prefixes: 09,002
RCF|10.1.1.10:1720|800:dialedDigits=Wei:h323_ID|terminal|1289_endp
Wed, 26 Jun 2002 16:40:55 +0800 C(0/32/39) <1>
RCF|10.0.1.66:1720|716:dialedDigits=Vicky:h323_ID|terminal|1425_endp
Wed, 26 Jun 2002 16:40:58 +0800 C(1/47/53) <1>
Number of Endpoints: 2
;
```

- PrintCurrentCalls, c, ! Muestra todas las llamadas acuales utilizando la misma sintaxis de ACF como en el establecimiento de llamada.

  Formato:

```
CurrentCalls
Call No. # | CallID | Call_Duration | Left_Time
Dialed_Number
ACF|Caller_IP:Port|Caller_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
ACF|Callee_IP:Port|Callee_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
...
Number of Calls: Current_Call Active: Active_Call From Neighbor: Call_From_Neighbor \
From Parent: Call_From_Parent
;
```

  Ejemplo:

```
CurrentCalls
Call No. 29 | CallID bd c6 17 ff aa ea 18 10 85 95 44 45 53 54 77 77 | 109 | 491
Dial 0953378875:dialedDigits
ACF|10.0.1.49:1720|4048_CGK1|25263|frank:h323_ID|gunter:h323_ID|false;
ACF|10.1.1.1:1720|4037_CGK1|25263|gunter:h323_ID|frank:h323_ID|true;
Call No. 30 | CallID 70 0e dd c0 9a cf 11 5e 00 01 00 05 5d f9 28 4d | 37 | 563
Dial 0938736860:dialedDigits
ACF|10.0.1.48:1032|4041_CGK1|11896|sue:h323_ID|peter:h323_ID|false;
ACF|10.1.1.1:1720|4037_CGK1|11896|peter:h323_ID|sue:h323_ID|true;
Number of Calls: 2 Active: 2 From Neighbor: 0 From Parent: 0
;
```

- PrintCurrentCallsVerbose, cv, !! Muestra el detalle de todas las llamadas actuales.

  Formato:

```
CurrentCalls
Call No. # | CallID | Call_Duration | Left_Time
Dialed_Number
ACF|Caller_IP:Port|Caller_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
ACF|Callee_IP:Port|Callee_EPID|CRV|DestinationInfo|SrcInfo|IsAnswered;
# Caller_Aliases|Callee_Aliases|Bandwidth|Connected_Time <r>
...
Number of Calls: Current_Call Active: Active_Call From NB: Call_From_Neighbor
;
```

  Ejemplo:

```
CurrentCalls
Call No. 48 | CallID 7d 5a f1 0a ad ea 18 10 89 16 00 50 fc 3f 0c f5 | 30 | 570
Dial 0225067272:dialedDigits
ACF|10.0.1.200:1720|1448_endp|19618|frank:h323_ID|gunter:h323_ID|false;
```

```
ACF|10.0.1.7:1720|1325_endp|19618|gunter:h323_ID|frank:h323_ID|true;
# Sherry:h323_ID|Accel-GW1:h323_ID|200000|Wed, 26 Jun 2002 17:29:55 +0800 <2>
Number of Calls: 1 Active: 1 From NB: 0
;
```

- Find, f Busca un endpoint registrado mediante el alias o un prefijo.

  Formato:

  ```
  Find Alias
  RCF|IP:Port|Aliases|Terminal_Type|EndpointID
  ;
  ```

  Ejemplo:

  ```
  f 800
  RCF|10.1.1.10:1720|800:dialedDigits=Wei:h323_ID|terminal|1289_endp
  ;
  f 801
  SoftPBX: alias 801 not found!
  ```

- FindVerbose, fv Busca el detalle de un endpoint registrado mediante un alias o un prefijo.

  Formato:

  ```
  FindVerbose Alias
  RCF|IP:Port|Aliases|Terminal_Type|EndpointID
  Registration_Time C(Active_Call/Connected_Call/Total_Call) <r>
  [Prefixes: ##] (gateway only)
  ;
  ```

  Ejemplo:

  ```
  fv 02
  RCF|10.0.1.100:1720|TFN:h323_ID|gateway|4037_CGK1
  Wed, 26 Jun 2002 17:47:29 +0800 C(0/84/120) <1>
  Prefixes: 02,09
  ;
  ```

- UnregisterIP Obliga a desregistrar un endpoint mediante su dirección IP y su puerto de señalización de llamada.

  Formato:

  ```
  UnregisterIP IP[:Port]
  ```

  Ejemplo:

  ```
  UnregisterIP 10.0.1.31:1720
  URQ|10.0.1.31:1032|1326_endp|maintenance;
  SoftPBX: Endpoint 10.0.1.31:1720 unregistered!
  ```

- UnregisterAlias Obliga a desregistrar un endpoint mediante uno de sus alias.

  Formato:

  ```
  UnregisterAlias Alias
  ```

  Ejemplo:

  ```
  UnregisterAlias 601
  URQ|10.0.1.31:1032|1326_endp|maintenance;
  SoftPBX: Endpoint 601 unregistered!
  ```

- UnregisterAllEndpoints Obliga a desregistrar todos los endpoints registrados.

  Formato:

  Ejemplo:

  ```
  UnregisterAllEndpoints
  URQ|10.0.1.7:1024|1325_endp|maintenance;
  URQ|10.0.1.8:1024|1322_endp|maintenance;
  URQ|10.0.1.32:1032|1324_endp|maintenance;
  URQ|10.0.1.36:1032|1323_endp|maintenance;
  URQ|10.0.1.42:1032|1318_endp|maintenance;
  Done
  ;
  ```

- DisconnectCall Desconecta una llamada mediante su número.

  Formato:

  ```
  DisconnectCall Number
  ```

  Ejemplo:

  ```
  DisconnectCall 1533
  ```

- DisconnectIP Desconecta todas las llamadas de un endpoint mediante su IP y su puerto de señalización de llamada.

  Formato:

  ```
  DisconnectIP IP[:Port]
  ```

  Ejemplo:

  ```
  DisconnectIP 10.0.1.31:1720
  ```

- DisconnectAlias Desconecta todas las llamadas de un endpoint mediante uno de sus alias.

  Formato:

  ```
  DisconnectAlias Alias
  ```

  Ejemplo:

  ```
  DisconnectAlias 601
  ```

- ClearCalls Desconecta todas las llamadas existentes en el gatekeeper.
- GK Muestra información del gatekeeper padre (parent gatekeeper).
- Trace Establece el nivel de rastreo de salida para la interfaz de estado. Este controla que mensajes son enviados a este cliente:
  - trace 0 o trace min Solamente respuestas directas a comandos y notificaciones de recarga (reload).
  - trace 1 CDRs, respuestas directas a comandos y notificaciones de recarga (reload).
  - trace 2 or trace max Muestra todo (RAS, CDRs, respuestas directas a comandos y notificaciones de recarga (reload), etc).
- Debug Solamente utilizado para propósito de debug. Opciones:

- trc [+|-|n] Muestra/modifica el nivel de rastreo.
- cfg SEC PAR Lee e muestra un parámetro de configuración de una sección.
- set SEC PAR VAL Escribe un valor a un parámetro de configuración de una sección.
- remove SEC PAR Remueve el valor de un parámetro de configuración de una sección.
- remove SEC Remueve una sección.
- printrm VERBOSE Muestra todos los registros de un endpoint removido.

Ejemplos:

```
debug trc 3
debug set RoutedMode H245Routed 1
```

- Who Muestra todos las personas que se encuentran en el puerto de estado.
- RouteReject Termina esta llamada sobre una cola virtual. Este comando es utilizado como respuesta a un evento RouteRequest (ver más adelante).

  Formato:

  ```
  RouteReject CallingEndpointID CallRef
  ```

  Ejemplo:

  ```
  RouteReject endp_4711 1234
  ```

- RouteToAlias, rta Enruta esta llamada sobre una cola virtual hacia el alias especificado. Este comando es utilizado como respuesta a un evento RouteRequest (ver más adelante).

  Formato:

  ```
  RouteToAlias Alias CallingEndpointID CallRef
  ```

  Example:

  ```
  RouteToAlias Suzi endp_4711 1234
  ```

- RouteToGateway, rtg Enruta esta llamada sobre una cola virtual hacia un alias especificado y establece el destinationSignallAddress. Este comando es utilizado como una respuesta al evento RouteRequest (ver más adelante). Usted puede utilizar este comando para rutear llamadas hacia gateways fuera de la zona (out-of-zone) o MCUs no registrados con el gatekeeper. Asegúese que las políticas 'vqueue' y 'explicit' esten en efecto para esas llamadas.

  Formato:

  ```
  RouteToGateway Alias IP:Port CallingEndpointID CallRef
  ```

  Ejemplo:

  ```
  RouteToGateway Suzi 192.168.0.50 endp_4711 1234
  ```

- Exit, q Cierra el puerto de estado.
- TransferCall Trasfiere una llamada establecida desde el alias A hacia el alias B. En un momento dado el alias A esta hablandocon el alias X, entonces el alias A eta hablando con el alias B después del TransferCall.

  Actualmente éste trabaja solamente con endpoints que soportan mensajes Q.931 Facility (por consiguiente éste comando no funciona con Netmeeting).

  Formato:

  ```
  TransferCall Source-Alias New-Destination-Alias
  ```

  Ejemplo:

  ```
  TransferCall Frank Peter
  ```

12.3. Mensajes (Referencia)

Esta sección describe los mensajes de salida hacia la interfaz de estado.

- GCF|IP|Aliases|Endpoint_Type; El gatekeeper recibe un GatekeeperRequest (GRQ) y responde con un GatekeeperConfirm (GCF).
- GRJ|IP|Aliases|Endpoint_Type|RejectReason; El gatekeeper recibe un GatekeeperRequest (GRQ) y responde con un GatekeeperReject (GRJ).
- RCF|IP:Port|Aliases|Endpoint_Type|EndpointID; El gatekeeper recibe un RegistrationRequest (RRQ) y responde con un RegistrationConfirm (RCF).
- RRJ|IP|Aliases|Endpoint_Type|RejectReason; El gatekeeper recibe un RegistrationRequest (RRQ) y responde con a RegistrationReject (RRJ).
- ACF|Caller_IP:Port|Caller_EndpointID|CRV|DestinationInfo|SrcInfo|IsAnswered; El gatekeeper recibe un AdmissionRequest (ARQ) y responde con un AdmissionConfirm (ACF).
- ARJ|Caller_IP:Port|DestinationInfo|SrcInfo|IsAnswered|RejectReason; El gatekeeper recibe un AdmissionRequest (ARQ) y responde con un AdmissionReject (ARJ).
- DCF|IP|EndpointID|CRV|DisengageReason; El gatekeeper recibe un DisengageRequest (DRQ) y responde con un DisengageConfirm (DCF).
- DRJ|IP|EndpointID|CRV|RejectReason; El gatekeeper recibe un DisengageRequest (DRQ) y responde con un DisengageReject (DRJ).
- LCF|IP|EndpointID|DestinationInfo|SrcInfo; El gatekeeper recibe un LocationRequest (LRQ) y responde con un LocationConfirm (LCF).
- LRJ|IP|DestinationInfo|SrcInfo|RejectReason; El gatekeeper recibe un LocationRequest (LRQ) y responde con un LocationReject (LRJ).
- BCF|IP|EndpointID|Bandwidth; El gatekeeper recibe un BandwidthRequest (BRQ) y responde con un BandwidthConfirm (BCF).
- BRJ|IP|EndpointID|Bandwidth|RejectReason; El gatekeeper recibe un BandwidthRequest (BRQ) y responde con un BandwidthReject (BRJ).
- UCF|IP|EndpointID; El gatekeeper recibe un UnregistrationRequest (URQ) y responde con un UnregistrationConfirm (UCF).
- URJ|IP|EndpointID|RejectReason; El gatekeeper recibe un UnregistrationRequest (URQ) y responde con un UnregistrationReject (URJ).
- IRQ|IP:Port|EndpointID; El gatekeeper envía un InfoRequest (IRQ) hacia un endpoint para consultarle si éste esta aún en actividad. El endpoint debe responder con un InfoRequestResponse (IRR) inmediatamente.
- URQ|IP:Port|EndpointID|Reason; El gatekeeper envía un UnregistrationRequest (URQ) hacia un endpoint para cancelar su registro. El endpoint debe responder con un UnregistrationConfirm (UCF).
- CDR|CallNo|CallId|Duration|Starttime|Endtime|CallerIP|CallerEndId| \ CalledIP|CalledEndId|DestinationInfo|SrcInfo|GatekeeperID; Después de una llamada desconectada, el registro de detalle de llamada (call detail record) es mostrado (en una línea).

- RouteRequest|CallerIP:Port|CallerEndpointId|CallRef|VirtualQueue|CallerAlias; Petición para que una aplicación externa enrute una llamada entrante sobre una cola virtual. Esto puede ser hecho con los comandos RouteToAlias o RouteReject.